Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Aplikace Android - teorie a praxe

Michal Bečka

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Michal Bečka

Informatika

Název práce

Aplikace Android – teorie a praxe

Název anglicky

Android application – theory and practice

Cíle práce

Hlavním cílem práce je charakteristika vývoje aplikací pro operační systém Android od společnosti Google, jejich distribuce a uplatnění na trhu.
Dílčí cíle diplomové práce jsou:
- analyzovat obecné požadavky na aplikace pro operační systém Android
- charakterizovat různe pohledy na využití operačního systému Android
- implementace vlastní aplikace a její uvedení na trh
- srovnání možností operačního systému Android s konkurencí (iOS, Windows Phone, J2ME)

Metodika

Metodika řešené problematiky diplomové práce je založena na studiu a analýze odborných informačních zdrojů. Vlastní řešení je realizováno formou návrhu a implementace vlastní jednoduché aplikace pro operační systém Android. Na základě syntézy teoretických poznatků a výsledků vlastního řešení budou formulovány závěry diplomové práce.

Doporučený rozsah práce

60 – 80 stran

Klíčová slova

Android, mobilní operační systém, Java

Doporučené zdroje informací

MEIER R. Professional Android Application Development, 2. vydání. Indianapolis: Wrox. 2010. 576 str. ISBN 0470565527.

MURPHY, M. L. Android 2, Průvodce programováním mobilních aplikací. Brno: Computer Press. 2011. 369 str. ISBN 978-80-251-3194-7.

ROGERS, R., LOMBARDO, J. Android Application Development, 1st Edition. Cambridge: O'Reilly Media. 2009. 336str. ISBN 978-0-596-52147-9.

TOPLEY, K. J2ME v kostce – Pohotová referenční příručka. Praha: GRADA. 2004. 536str. ISBN 80-247-0426-9.

Předběžný termín obhajoby

2016/17 LS – PEF

Vedoucí práce

Ing. Čestmír Halbich, CSc.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 18. 10. 2016

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 24. 10. 2016

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 26. 01. 2017

Čestné prohlášení

Prohlašuji, že svou diplomovou práci „Aplikace Android - teorie a praxe“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dašich informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 15. března 2017 .....

Poděkování

Chtěl bych poděkovat především vedoucímu této diplomové práce Ing. Čestmíru Halbichovi, CSc. za jeho podporu a konzultace při vedení práce. Dále bych rád poděkoval své rodině, která mne po celou dobu studia podporovala. Mé poděkování také patří všem přátelům a známým, kteří zapůjčili své telefony pro testování aplikace vyvinuté v praktické části.

Aplikace Android - teorie a praxe

Android application - theory and practice

Souhrn

Tato práce se zabývá srovnáním mobilních operačních systémů spolu s popisem možností a architektury mobilního operačního systému Android. Praktická část této práce obsahuje analýzu, vývoj, testování a publikaci aplikace pro mobilní operační systém Android. Vyvíjená aplikace si klade za cíl umožnit uživateli zobrazovat běžně nepřístupná data z vybraných senzorů telefonu, informace o stavu Wi-Fi připojení, mobilní sítě a baterie.

Klíčová slova Android, mobilní operační systém, Java, Google, Google Play, Android SDK, Eclipse

Summary

This thesis contains comparison of mobile operating systems and the overview of the possibilities and of the architecture of mobile operating system Android. The practical part of this thesis includes analysis, implementation, testing and publishing of application for mobile operating system Android. The developed application aims to allow users to display usually inaccessible data from certain sensors of the mobile phone, information about Wi-Fi and cell connection state as well as detailed information about battery state.

Keywords Android, mobile operating system, Java, Google, Google Play, Android SDK, Eclipse

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

První skutečně přenosné mobilní telefony se objevily na trhu v sedmdesátých letech minulého století. Díky postupnému procesu miniaturizace dosáhla v devadesátých letech velikost mobilních telefonů kapesních rozměrů. Z důvodu nutnosti snadného ovládání prsty a dobré viditelnosti displeje již další zmenšování mobilních telefonů nebylo žádoucí. Od tohoto momentu se hlavním cílem inovace mobilních telefonů stalo rozšiřování stávajících funkcí telefonu.

Mobilní telefony získaly schopnost zobrazení času, nastavení budíku, kalendáře, kalkulačky a mnoho dalších funkcí užitečných v běžném životě. Následným krokem evoluce bylo nahrazení monochromatického displeje displejem barevným a přidání fotoaparátu do většiny telefonů.

Na přelomu tisíciletí výkon mobilních telefonů narostl tolik, že začal umožňovat tvorbu složitějších specializovaných aplikací pro mobilní telefony. Prvním široce rozšířeným standardem pro vývoj mobilních aplikací se stala v roce 2000 specifikace J2ME. Ta umožňovala tvorbu malých aplikací, které byly psány v jazyce Java. Nevýhodou platformy J2ME byl ovšem velmi komplikovaný způsob distribuce a prodeje aplikací spojený kombinací mnohých omezení, která se lišila mezi jednotlivými výrobci telefonů.

Ačkoliv mobilní telefony s operačním systémem existovaly již dříve, k jejich skutečně masovému rozšíření došlo od roku 2007, kdy společnost Apple uvedla na trh svůj telefon iPhone. První mobilní telefon s OS Android byl uveden na trh o rok později. Tyto dvě platformy vyřešily většinu problémů, se kterými se potýkaly aplikace platformy J2ME. Telefony s OS Android díky příznivější cenové politice předstihly v celkovém počtu uživatelů telefony společnosti Apple. Nyní je OS Android nejrozšířenějším mobilním operačním systémem.

Mobilní telefony s OS Android obsahují řadu senzorů, avšak samotný Android nedokáže uživateli zobrazit veškerá dostupná data z těchto senzorů. Tato diplomová práce je zaměřena na možnosti získání podrobných dat z senzorů telefonu a informací o stavu telefonu obecně. A na následné přehledné zobrazení těchto dat uživateli v aplikaci, která je součástí praktické části této práce.

Kapitola 2

Cíl práce a metodika

2.1 Cíl

Hlavním cílem práce je charakteristika vývoje aplikací pro operační systém Android od společnosti Google, jejich distribuce a uplatnění na trhu.

Dílčí cíle diplomové práce jsou:

- Analyzovat obecné požadavky na aplikace pro operační systém Android.
- Charakterizovat různé pohledy na využití operačního systému Android.
- Implementace vlastní aplikace a její uvedení na trh.
- Srovnání možností operačního systému Android s konkurencí (iOS, Windows Phone, J2ME).

2.2 Metodika

Metodika řešené problematiky diplomové práce je založena na studiu a analýze odborných informačních zdrojů. Vlastní řešení je realizováno formou návrhu a implementace vlastní jednoduché aplikace pro operační systém Android. Na základě syntézy teoretických poznatků a výsledků vlastního řešení budou formulovány závěry diplomové práce.

Kapitola 3

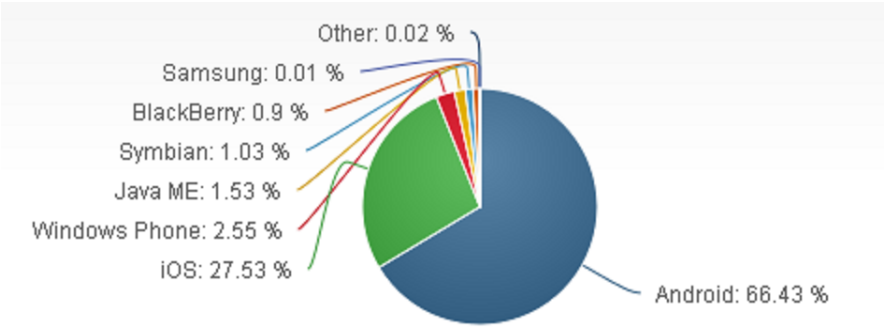
Charakteristika vývoje aplikací pro OS Android

Dnes je mobilní operační systém Android světově nejrozšířenějším mobilním operačním systémem (35). Tomuto dnešnímu stavu ovšem předcházela velmi dlouhá cesta, jejíž počátky sahají až k prvním osobním počítačům, od kterých pokračuje přes první skutečně přenosné mobilní telefony, které se na trhu objevily v sedmdesátých letech minulého století.

První mobilní telefony byly určeny pouze k jednomu primárnímu účelu – telefonování. Avšak postupem času získávaly novější modely další přidané funkce. Až jejich vývoj v roce 2000 dosáhl takové úrovně, že byla vytvořena specifikace J2ME, jakožto standard umožňující tvorbu jednoduchých aplikací vývojářů třetích stran pro mobilní telefony. Od platformy J2ME již byl třeba pouze jeden vývojový krok k vytvoření mobilních telefonů s vlastním operačním systémem. Mobilní operační systém Android nebyl rozhodně prvním mobilním operačním systémem, avšak svou praktičností a cenovou politikou si vydobyl své nynější místo na trhu mobilních operačních systémů. Přehled nejvýznamnějších konkurenčních mobilních operačních systémů je uveden v části č. 3.1.

Samotný mobilní operační systém Android si během svého vývoje, jehož počátky sahají až do roku 2003, prošel mnoha vývojovými fázemi, během kterých vznikla celá řada verzí tohoto operačního systému. Samotný systém Android byl navržen, jako vícevrstvý systém. Jeho jádro je odvozeno z jádra operačního systému Linux. Nad samotným jádrem jsou umístěny další vrstvy, které zajišťují funkčnost jednotlivých částí systému. Vývojové fáze spolu s významnými změnami a s architekturou samotného systému jsou podrobně popsány v části č. 3.2.

Hlavní výhodou telefonů s operačním systémem je možnost spouštět aplikace vývojářů třetích stran. Společnost Google Inc. je si tohoto aspektu velmi dobře vědoma, a proto se snaží podporovat vývojáře aplikací, aby samotný proces vývoje byl pokud možno přívětivý, a co nejméně časově náročný. Zároveň



Obrázek 3.1: Podíl mobilních OS na trhu v srpnu 2016 (Zdroj: (35))

společnost Google Inc. spravuje svůj internetový obchod s aplikacemi pro zařízení s operačním systémem Android – Google Play. V tomto internetovém obchodě umožňuje vývojářům aplikací různé modely zpeněžení vystavených aplikací, které jsou k dispozici uživatelům. Jednotlivé aspekty vývoje aplikací pro platformu Android jsou rozebrány v částech č. 3.3 až č. 3.5. Samotné možnosti šíření a zpeněžení vytvořené aplikace jsou popsány v částech č. 3.6 a č. 3.7.

3.1 Mobilní operační systémy

Dle údajů OSN žilo v srpnu 2016 na Zemi více, než 7,4 miliardy obyvatel (38). Ze statistik uvedených na statisticky zaměřeném serveru statista.com vyplývá, že celkový počet aktivních mobilních telefonů je srovnatelný s celkovým počtem obyvatel planety Země. Dalším zajímavým údajem je skutečnost, že již více, jak 2 miliardy uživatelů mobilních telefonů používají mobilní telefon s některým z operačních systémů (42). Z predikce dále vyplývá, že trend celkového počtu uživatelů mobilních telefonů s operačním systémem bude rostoucí. Z tohoto pohledu se jedná o velmi dynamický a rozvíjející se trh, kde vzniká mnoho příležitostí pro vývoj a prosazení řady nových mobilních aplikací a konceptů.

V dnešní době je většina trhu mobilních telefonů s operačním systémem rozdělena mezi tři hlavní platformy tří velkých společností. A to konkrétně nejrozšířenější platforma Android od společnosti Google Inc., iOS od společnosti Apple Inc. a Windows Phone (Windows Mobile) od společnosti Microsoft Corporation.

Cesta k dnešnímu stavu na trhu ovšem nebyla tak přímočará a předcházelo mu několik dnes již velmi málo používaných mobilních operačních systémů a platforem. Které ovšem více, či méně, ovlivnily podobu dnešních nejrozšířenějších mobilních operačních systémů. V této části jsou proto popisovány

nejvýznamnější předchůdci a konkurenti dnešních nejrozšířenějších mobilních operačních systémů. Není zde již dále popisován operační systém Android od společnosti Google Inc., neboť je mu dále věnována kompletní samostatná část.

3.1.1 Platforma J2ME

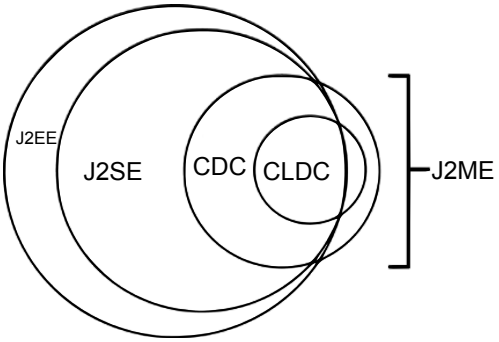
Při sestavování informací o platformě J2ME (Java 2 Micro Edition) jsem vycházel ze své bakalářské práce(1), kde se tomuto tématu podrobně věnuji, dále z materiálů dostupných na portále společnosti Oracle (36) a z informací, které uvádí Kim Topley ve své publikaci J2ME v kostce (45).

Java 2 Micro Edition (dále jen J2ME) není mobilním operačním systémem, ale platformou, která unifikuje způsob tvorby, distribuce a spouštění mobilních aplikací. Svými vlastnostmi se proto v některých ohledech podstatně odlišuje od zbylých mobilních operačních systémů uvedených v této části. J2ME ovšem položila základní kameny možností tvorby uživatelských aplikací pro mobilní telefony, na které dále navazují mobilní telefony s operačním systémem, a proto je v této části uvedena také.

Java 2 Micro Edition (dále jen J2ME) je jednou z několika edic jazyka Java. Zatímco J2SE (Java 2 Standard Edition) je zaměřena na využití v oblasti osobních počítačů a J2EE (Java 2 Enterprise Edition) je určena pro použití v podnikových aplikacích a informačních systémech, J2ME byla navržena pro použití v zařízeních s omezenými prostředky. Konkrétně mobilních telefonech, PDA, set-top boxech, některých herních konzolích a elektronických hračkách. Největšího rozšíření J2ME ovšem dosáhla právě v oblasti mobilních telefonů.

J2ME byla hierarchicky rozčleněna podle účelu jejího využití pro různé druhy zařízení. Toto členění bylo provedeno pomocí tzv. konfigurací. Dodatečné požadavky na další funkce v rámci dané konfigurace byly specifikovány pomocí tzv. profilů. Aplikace pro mobilní telefony byly specifikovány pomocí konfigurace CLDC (Connected Limited Device Configuration) a profilu MIDP (Mobile Information Device Profile). Aplikace, které využívaly konfiguraci CLDC a profil MIDP byly nazývány, jako tzv. MIDlety.

Historie platformy J2ME se začala psát v roce 1998, kdy společnost Sun Microsystems zahájila vývoj platformy PersonalJava pro nově se rozvíjející trh mobilních zařízení. V roce 2000 byla částečně nevyhovující specifikace PersonalJava nahrazena specifikací J2ME. Postupem času byly přidávány nové konfigurace a profily. Počátkem roku 2003 byla přidána konfigurace CLDC 1.1, která rušila mnohá původní omezení. Například umožňila použití datových typů s plovoucí desetinnou čárkou, či slabých referencí. V roce 2003 byl uveden také profil MIDP 2.0, který nově umožňoval práci s multimédii a přidal možnost trvalého uložení uživatelských dat. Poslední větší novinkou bylo uvedení profilu MIDP 3.0 v roce 2009, který umožňuje spuštění několika aplikací zároveň, či automatický start aplikace na pozadí. Do dnešních dnů však bylo vydáno pouze několik málo mobilních telefonů podporujících profil MIDP 3.0.



Obrázek 3.2: Diagram průniku společných knihoven Java(Zdroj: (5))

3.1.2 Symbian

Počátky vývoje operačního systému Symbian se datují až do 80. let 20. století, kdy začal být vyvíjen společností Psion pod starším názvem EPOC. Ke skutečnému vzniku názvu Symbian došlo v roce 1998, kdy společnosti Ericsson, Nokia a Psion založily novou společnost Symbian Ltd., jejímž cílem byl další vývoj operačního systému Symbian. Nejširší využití ovšem Symbian našel na mobilních telefonech značky Nokia.

Symbian byl naprogramován v jazyce C++ a umožňoval spouštění uživatelských aplikací naprogramovaných také v jazyce C++ spolu s využitím multiplatfrormní grafické knihovny Qt. Zajímavostí ovšem je, že novější verze operačního systému Symbian byly schopné spouštět i aplikace, které byly vytvořeny pro platformu J2ME (ovšem za cenu částečného propadu výkonu oproti nativním Symbian aplikacím).

Symbian ovšem trpěl zpětnou nekompatibilitou aplikací mezi některými verzemi. Konkrétně například populární Symbian 9.1 neumožňoval bez předchozí úpravy spuštění aplikací a her určených pro starší verze tohoto OS.

Na přelomu let 2011 a 2012 ovšem společnost Nokia rozhodla ukončit další vývoj systému Symbian. Pro další telefony s operačním systémem se Nokia zavázala využívat mobilní operační systémy od společnosti Microsoft, konkrétně v té době Windows Phone 7 a jeho další verze.

3.1.3 BlackBerry OS

BlackBerry OS je mobilní operační systém, který je zaměřen téměř výhradně na korporátní klientelu. Tento operační systém byl vyvinut společností Research In Motion Limited, která se v roce 2013 přejmenovala na BlackBerry Limited (7). BlackBerry OS je součástí celkového řešení, které se dále skládá z mobilních telefonů BlackBerry s tímto operačním systémem a přidruženého cloudového řešení, které umožňuje bezpečnou šifrovanou komunikaci. Přidružené cloudové řešení nabízí dva možné druhy použití:

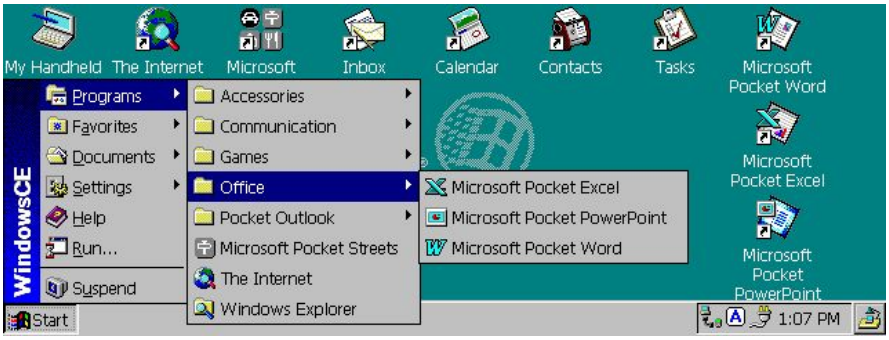
Obrázek 3.3: Uživatelské rozhraní BlackBerry OS verze 4.6(Zdroj: (9))

- BIS (BlackBerry Internet Service) je řešení určené pro menší firemní zákazníky, kdy společnost BlackBerry Limited poskytuje cloud, jako službu. Zákazník se tedy nemusí starat o údržbu samotného cloudového řešení.
- BES (BlackBerry Enterprise Server) je řešení určené pro velké korporátní zákazníky, kdy je cloudové řešení nainstalováno na server zákazníka, a ten má tak veškerá data naprosto pod kontrolou.

Blackberry OS umožňuje spouštění J2ME aplikací. Z důvodu bezpečnosti BlackBerry Limited poskytuje svá vlastní rozšíření specifikace J2ME, která umožňují vývojářům aplikací dosáhnout vyšší bezpečnosti komunikace a grafické a funkční začlenění aplikace do samotného BlackBerry OS (37). BlackBerry OS od verze 10.3 umožňuje i instalaci nativních Android aplikací(6). Díky tomuto rozšíření uživatelé BlackBerry OS získali přístup k aplikacím vytvořeným primárně pro OS Android.

3.1.4 Microsoft Windows

Společnost Microsoft vydala svůj první čistě mobilní operační systém již v roce 1996. Jednalo se o modulární mobilní operační systém Windows CE. Jeho uživatelské rozhraní bylo navrženo převážně na ovládání pomocí stylusu, neboť některé prvky uživatelského rozhraní byly příliš malé na možnost přesného stisknutí pomocí prstu. Celkový způsob používání byl tedy svým konceptem velmi odlišný od způsobu ovládání dnešních mobilních operačních systémů.



Obrázek 3.4: Uživatelské rozhraní Windows CE(Zdroj: (10))

Windows CE byl primárně určen pro PDA a podobná zařízení, mobilní telefony nebyly ještě hlavní cílovou skupinou tohoto mobilního operačního systému.(25)

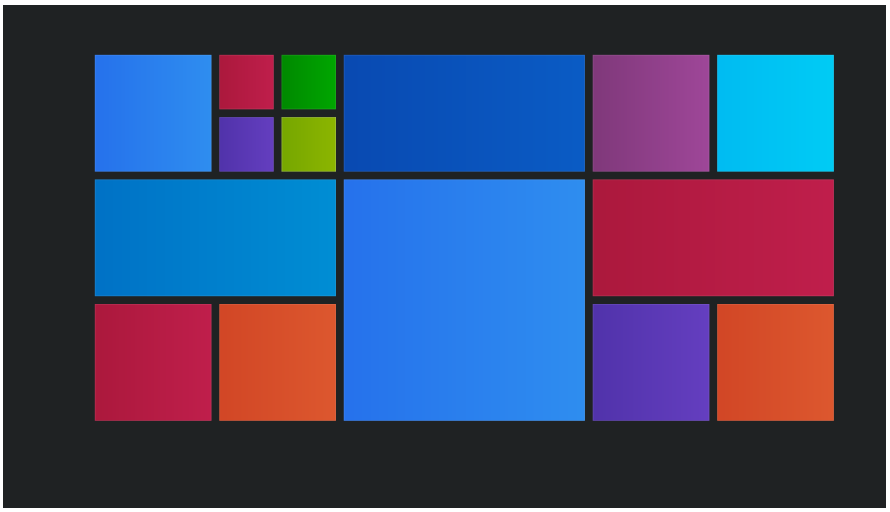
V roce 2000 Microsoft představil mobilní operační systém PocketPC 2000, který byl následně v roce 2003 přejmenován na Windows Mobile. Jednalo se o první mobilní operační systém, který byl společností Microsoft cílen primárně na mobilní telefony. V roce 2007 se Windows Mobile stal nejpoužívanějším mobilním operačním systémem ve Spojených státech amerických.(11). Svým návrhem uživatelského rozhraní a svými schopnostmi ovšem Windows Mobile silně zaostával za nově uvedeným iOS od společnosti Apple a právě za OS Android od společnosti Google. Aby Microsoft byl schopen konkurovat těmto dvěma novým a dynamicky se rozvíjejícím platformám, musel učinit rychlé rozhodnutí. K tomuto rozhodnutí došlo v roce 2010, kdy byl oznámen konec vývoje platformy Windows Mobile, a zároveň bylo oznámeno uvedení nové mobilní platformy Windows Phone.

Uživatelské rozhraní Windows Phone je navrženo tak, aby respektovalo nové trendy ve způsobu ovládání mobilních aplikací, které přinesly operační systémy iOS a Android. Zároveň však Windows Phone přinesl nové uživatelské rozhraní nazvané Metro (Metro bylo z licenčních důvodů v roce 2012 přejmenováno na Modern UI (44)).

Modern UI (dříve Metro)

Modern UI používá, jako základní stavební kámen uživatelského rozhraní tzv. dlaždice. Dlaždice mohou plnit více funkcí. Primárně plní funkci ikony aplikace, která slouží k jejímu spuštění. Dále ovšem dlaždice velmi často plní funkci podobnou widgetu u OS Android. Dokáží tedy zobrazovat informace (například informace o počasí, aktuální dopravní informace, nebo počet nepřečtených zpráv). Dlaždice mohou nabývat různých velikostí. Menší rozměr je ovšem vždy možné získat čtvrcením, nebo půlením větší dlaždice. Microsoft se snaží zavést Modern UI i ve svých desktopových operačních systémech Windows 8 a Windows 10.

Obrázek 3.5: Uživatelské rozhraní Windows Mobile 6.1(Zdroj: (31))



Obrázek 3.6: Prázdné dlaždice rozhraní Modern UI(Zdroj:Autor)

Windows Phone 7

Windows Phone 7 byl uveden na trh na podzim roku 2010. Operační systém používal jádro z Windows CE 7 (32), což vedlo k mnohým problémům s následnou binární nekompatibilitou aplikací, které byly vyvinuty pro Windows Phone 7. Stejně tak telefony, které byly vydány s Windows Phone 7, nebylo možné upgradovat na novější verzi Windows Phone 8 (39). Toto byl důsledek velmi rychlého vývoje platformy, se kterou se Microsoft snažil získat podíl na nově vzniklém silně dynamickém trhu, kde již velkou většinu získávaly společnosti Apple a Google.

Aplikace pro Windows Phone 7 byly typicky vytvářeny prostřednictvím platformy Silverlight, nebo XNA frameworku (nikoliv přímo v jazyce C++, nebo C#).

Windows Phone 8

V říjnu 2012 Microsoft uvedl na trh Windows Phone 8, které již používají jádro Windows NT (tedy shodné, jako desktopové operační systémy od společnosti Microsoft). Microsoft rozšířil Windows Phone 8 o možnosti zpětné kompatibility s aplikacemi vytvořenými původně pro Windows Phone 7, čímž překlenul binární nekompatibilitu aplikací, která vznikla kvůli rozdílným jádrům operačního systému.

Na jaře 2014 Microsoft uvedl Windows Phone 8.1, které dokončily propojení vývoje aplikací pro Windows Phone a desktopové Windows 8.1 a vyšší. Od této verze je možné univerzální aplikace spouštět na mobilní i desktopové verzi operačního systému. Dodatečná funkčnost aplikace, která je využitelná pouze na jedné z platforem (například použití fotoaparátu) může být na druhé platformě skryta.

Aplikace pro Windows Phone 8 jsou opět vytvářeny v C++. Konkrétně C++11, což je částečně upravená verze C++, reflektující některé prvky jazyku Objective-C, který byl vyvinut společností Apple, a je využíván při vývoji pro platformu iOS (29).

Windows RT

Windows RT je dnes již dále nevyvíjená verze operačního systému, která byla určena pro mobilní zařízení s procesorem, který podporuje instrukční sadu ARM. Na tomto mobilním operačním systému nebylo z důvodu binární nekompatibility možné spouštět aplikace určené pro Windows Phone 8, nebo pro desktopové verze Windows (používající instrukční sadu x86 a x64). Windows RT byly navrženy po vzoru iOS, jako uzavřený operační systém, kam bylo možné instalovat pouze schválené aplikace z internetového obchodu společnosti Microsoft (Windows Store). Systém byl uveden na trh v roce 2012 spolu s tablety Microsoft Surface, na které byl primárně určený. Windows RT použí-

Obrázek 3.7: Uživatelské rozhraní Windows Phone 8.1(Zdroj: (43))

valy téměř shodné uživatelské rozhraní s Windows 8, což vedlo ke zmatení mnohých méně znalých uživatelů.

Ovšem z důvodu binární nekompatibility aplikací a uzavřenosti systému si Windows RT nezískalo oblibu mezi uživateli. Z tohoto důvodu v únoru 2015 Microsoft oznámil ukončení výroby tabletu Surface 2 a telefonu Lumia 2520, což byla poslední zařízení vyráběná s tímto operačním systémem. Tablet Surface 3 byl již navržen s podporou instrukční sady x86.

Windows 10 Mobile

Windows 10 Mobile, který navazuje na Windows Phone 8, byl oznámen v lednu roku 2015. Windows 10 Mobile používá, jako základ uživatelského rozhraní, také Modern UI, které oproti předchozím verzím získalo některá drobná vylepšení. Tento mobilní operační systém je součástí nové strategie společnosti Microsoft, kdy se společnost snaží o maximální možné sjednocení aplikací pro mobilní a desktopovou platformu. Označení Windows 10 Mobile si klade za cíl zdůraznit

Obrázek 3.8: Tablet Microsoft Surface 2(Zdroj: (28))

právě toto propojení s novým operačním systémem společnosti Microsoft pro desktop, a to s Windows 10.

Desktopový operační systém Windows 10 byl vydán v létě roku 2015. Společnost Microsoft umožnila po určité časové období uživatelům bezplatnou možnost aktualizace ze starších Windows 7 a Windows 8. S Windows 10 se rapidně mění filozofie vydávání nových verzí operačního systému. Místo vydávání celých nových verzí operačního systému společnost Microsoft plánuje pravidelné vydávání větších aktualizací (obdobu Service Packů, které byly vydávány pro starší verze jeho operačních systémů). V této filozofii je patrný příklon k osvědčenému způsobu vydávání pravidelných velkých aktualizací, který používá společnost Apple u svého desktopového operačního systému Mac OS X. U operačního systému Windows 10 Mobile se společnost Microsoft chce vydat obdobnou cestou.

Windows 10 Mobile zdůrazňuje použití tzv. Universal Windows Platform(30). Tato platforma umožňuje tvorbu aplikací tak, aby je bylo možné spustit na všech operačních systémech společnosti Microsoft z řady Windows 10 operačních systémů. A to pouze s minimálními úpravami ve zdrojovém kódu aplikace. Tento koncept například umožňuje připojení monitoru, myši a klávesnice k mobilnímu telefonu s Windows 10 Mobile a následné spuštění desktopového rozhraní aplikace z mobilního telefonu.

Pro Windows 10 Mobile zároveň byla společností Microsoft vyvinuta sada nástrojů, která má vývojářům mobilních aplikací umožnit snadné přenesení mobilní aplikace, která byla vytvořena původně pro platformu Android v jazyce Java, nebo pro platformu iOS v jazyce Objective-C (4).

Obrázek 3.9: Uživatelské rozhraní Windows 10 Mobile(Zdroj: (33))

3.1.5 Apple iOS

Společnost Apple Inc. pro svá mobilní zařízení používá svůj operační systém iOS. Tento operační systém je používaný výhradně na zařízeních prodávaných společností Apple Inc. a není poskytován pro použití žádným třetím stranám.

První verze mobilního operačního systému iOS byla vydána v roce 2007 spolu s uvedením prvního telefonu iPhone. Prvních několik verzí tohoto mobilního operačního systému bylo pojmenováno, jako iPhone OS. Až čtvrtá verze byla přejmenována na název iOS, kde se společnost Apple Inc. dohodla se společností Cisco Systems, Inc. o využití tohoto názvu, který byl společností Cisco použit dříve.

První verze iOS neobsahovala podporu spouštění aplikací třetích stran. Tu přinesla až druhá verze iOS, která byla vydána v roce 2008.

Do zařízení s mobilním operačním systémem iOS není oficiální cestou možné nahrávat aplikace třetích stran bez předchozího schválení společností Apple Inc. Všechny aplikace jsou distribuovány přes online obchod App Store, kde si společnost Apple Inc. vyhrazuje právo zamítnout aplikaci z objektivních důvodů, které popíše vývojáři, aby měl možnost sjednat jejich nápravu (toto se děje typicky během vývoje většiny aplikací). Zároveň si ovšem ponechává i možnost zamítnout vydání aplikace i bez udání důvodu a možnosti odvolání. Tento institut společnost Apple Inc. již v minulosti několikrát použila k nevpuštění nepohodlné konkurence do App Store. Například při sporu se společností Adobe Systems o možnost použití technologie Flash na zařízeních společnosti Apple Inc(41).

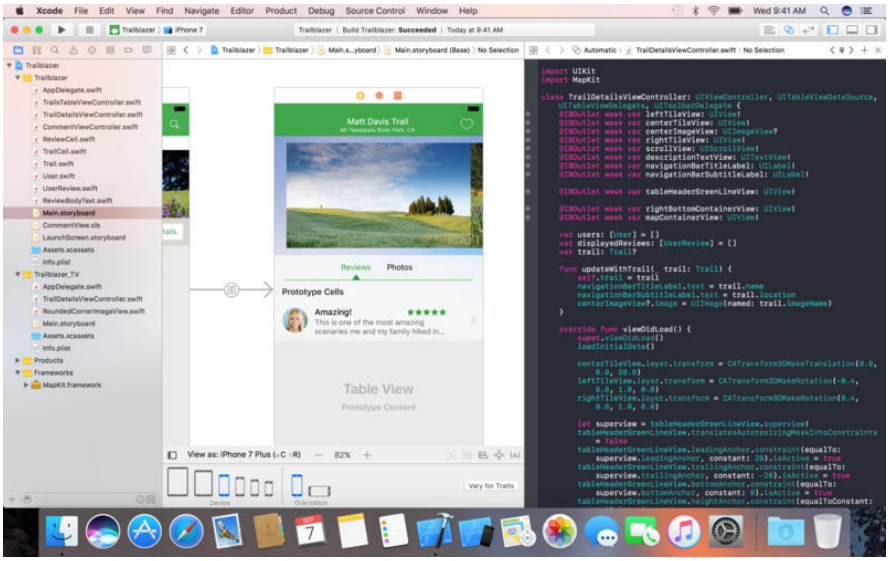
Postupem času vznikl ustálený princip, kdy společnost Apple Inc. před-

Obrázek 3.10: Telefony iPhone 1, 4, 5, 6 a 6 Plus(Zdroj: (8))

stavuje novou verzi mobilního operačního systému iOS vždy v září na konferenci Apple WWDC (Worldwide Developer Conference). Nová verze je vydána vždy během několika týdnů po představení. Aktuálně poslední verzí je iOS 10, který byl představen na Apple WWDC 2016. Tuto nejnovější verzi iOS 10 lze spustit na telefonech iPhone 5 a novějších.

Kontrola aplikací v App Store slouží zároveň k udržení kvality aplikací nabízených uživatelům. Neboť společnost Apple Inc. byla první, která si silně uvědomila význam optimalizace mobilních aplikací tak, aby používaly minimum operační paměti a používaly co nejméně často náročné přepínání kontextu na CPU (použití vícevláknového zpracování pouze tam, kde opravdu přináší výhody). Navíc společnost Apple trvá na tom, aby všechny aplikace uvedené v App Store používaly nativní kód (nikoliv interpretovaný bytecode apod.). Poslední velkou výhodou platformy iOS je tvorba aplikací s použitím ARC (Automatic Reference Counting), tedy s automatickým počítáním odkazů na objekty alokované v operační paměti telefonu (díky které není třeba používat k čištění operační paměti Garbage collection, jako je tomu u operačního systému Android). Všechny tyto vlastnosti iOS přispívají ke snaze společnosti Apple Inc. držet minimální energetickou náročnost mobilních aplikací, a tedy ke snaze o zvýšení výdrže mobilního telefonu.

Společnost Apple Inc. pečlivě hlídá celý životní proces vývoje a distribuce mobilní aplikace. A to tak, že oficiálně je možné mobilní aplikace vyvíjet pouze ve vývojovém prostředí Xcode, které je dostupné výhradně pro Mac OS X, který je možné spustit pouze na počítačích Macintosh, což je skutečnost, která odrazuje mnohé tvůrce mobilních aplikací od tvorby pro platformu iOS. Vývojové prostředí Xcode je ovšem zcela zdarma a je pro vývoj na danou platformu velmi intuitivní. Navíc Xcode obsahuje simulátor telefonů iPhone, který umožňuje rychlé testování vytvářené aplikace. Další komplikací při vývoji mobilních aplikací pro iOS je nutnost platby ročního poplatku 99 USD za vývojářský účet na App Store. Tento účet je naprosto nezbytný pro vývoj pro platformu iOS. Neboť i pro pouhé nahrání aplikace z vývojového prostředí Xcode



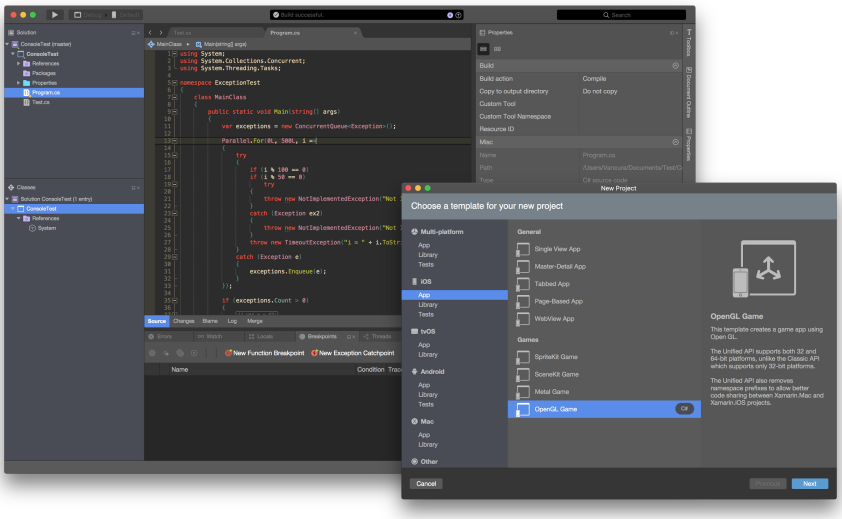
Obrázek 3.11: Vývojové prostředí Xcode(Zdroj: (3))

přes USB kabel je třeba ve vývojářské sekci App Store získat certifikát a tzv. Provisioning Profile, kterými je třeba aplikaci podepsat, aby ji bylo možné nahrát přes USB do iPhone. Dnes se pro distribuci testovacích verzí aplikace již ovšem častěji používá systém TestFlight, který je také součástí vývojářské sekci App Store. TestFlight umožňuje rozeslat pozvánku omezenému počtu uživatelů (testerů), kteří vždy po nahrání nové testovací verze aplikace do TestFlight obdrží notifikaci s odkazem na její stažení ve svém iPhone.

Po samotném otestování nastává proces odeslání aplikace ke schválení společností Apple Inc. pro uveřejnění v App Store. Tento proces dříve trval i týdny, s možností několika případných opakování v případě nalezených problémů ze strany kontroly společnosti Apple Inc. V roce 2016 došlo k částečnému urychlení tohoto procesu, kdy typicky trvá okolo jednoho týdne.

Každá další aktualizace aplikace musí opět projít schvalovacím procesem. V případě aktualizací ovšem kratším, který trvá typicky jednotky dnů. Toto je ovšem úzkým hrdlem při nutnosti schválit rychle nový build aplikace, který obsahuje opravu kritické chyby. Pro takové případy Apple poskytuje formulář s žádostí o tzv. Expedited review (rychlé schválení), kdy společnost Apple Inc. upřednostní schvalování daného buildu aplikace před ostatními aplikacemi. Tento formulář lze použít pouze několikrát do roka v dobře odůvodněných případech.

Neoficiální alternativou, která umožňuje vývoj mobilních aplikací pro iOS bez nutnosti vlastnit počítač Macintosh je vývojové multiplatformní prostředí Xamarin, které je v současné době společností Apple Inc. tolerováno. Tato tolerance pramení hlavně ze skutečnosti, že Xamarin vytváří buildy aplikací v nativním kódu(47). Pro tvorbu aplikací v prostředí Xamarin se používá



Obrázek 3.12: Vývojové prostředí Xamarin Studio(Zdroj: (48))

programovací jazyk C#.

Pro vývoj v prostředí Xcode byl dlouhou dobu využíván pouze programovací jazyk Objective-C. V roce 2014 byl na Apple WWDC představen nový programovací jazyk Swift, který si klade za cíl přinést příjemnější syntaxi (bližší syntaxi jazyka Java), a zrychlit a zpříjemnit tvorbu efektivního a udržitelného zdrojového kódu. Swift je ovšem dynamicky se vyvíjejícím jazykem, kde došlo k několika změnám syntaxe mezi jednotlivými verzemi jazyka. Společnost Apple Inc. vždy připravila nástroj, který byl schopen převést většinu kódu do novější verze jazyka. Ale i přes tuto podporu se přechod na novější verzi málo kdy obešel bez nutnosti ručního zásahu, korekcí a následné nutnosti testování celé aplikace. Proto mnoho vývojářů zůstává stále u jazyka Objective-C, dokud se překotný vývoj mladého jazyka Swift nezpomalí a syntaxe neustálí.

3.2 Mobilní operační systém Android

Při sestavování informací o mobilním operačním systému Android jsem vycházel částečně ze své bakalářské práce(1), kde se tomuto tématu také věnuji, z oficiálních stránek společnosti Google Inc., které popisují historii platformy Android(18), a z informací, které uvádí Mark Murphy ve své publikaci Android 2, Průvodce programováním mobilních aplikací (34).

Android je mobilní operační systém (dále pouze OS) pro přenosná zařízení založený na jádře operačního systému Linux verze 2.6 a novějších. Android je navržen pro použití v mobilních telefonech, tabletech, navigacích a dalších mobilních zařízeních. Dnes je Android nejrozšířenějším operačním systémem pro mobilní telefony. Konkrétně dle měření analytické společnosti

Obrázek 3.13: Telefon iPhone a tablet iPad(Zdroj: (12))

NET APPLICATIONS(35) 66,43% mobilních zařízení, která se v srpnu 2016 připojila k síti Internet, používalo OS Android.

3.2.1 Historie OS Android

Počátky historie OS Android sahají do října roku 2003, kdy se čtveřice Andy Rubin, Rich Miner, Nick Sears a Chris White rozhodla založit společnost, která si kladla za cíl tvorbu operačního systému pro mobilní zařízení. Původní primární cílovou skupinou byl hlavně nově se rozvíjející trh digitálních fotoaparátů. Až během vývoje bylo učiněno rozhodnutí, že nově vytvářený mobilní operační systém bude cílen na širší paletu mobilních zařízení, a nejvíce na mobilní telefony.

Do širšího povědomí se jméno společnosti Android, Inc. dostalo až v roce 2005, kdy byla koupena společností Google, Inc. V listopadu 2007 společnost Google Inc. iniciovala založení konsorcia Open Handset Alliance, jehož hlavním cílem je vývoj operačního systému Android. Mezi členy konsorcia patří například společnosti Google Inc., Intel Corporation, Samsung Electronics, nVidia Corporation a mnohé další. Členy konsorcia se stalo mnoho společností z řad mobilních operátorů, výrobců polovodičových součástek, výrobců mobilních

Obrázek 3.14: Telefon HTC Dream(Zdroj: (2))

zařízení a také z oblasti marketingu.

Na začátku listopadu 2007 také proběhlo oficiální představení OS Android veřejnosti spolu s představením první sady vývojových nástrojů (SDK). A byla nastíněna filozofie celé platformy, jako otevřená a komplexní platforma pro všechna představitelná mobilní zařízení, která obsahuje open-source operační systém, uživatelské rozhraní a aplikace které mohou vytvářet vývojáři třetích stran. A to vše bez omezení různými svazujícími patenty.

Na podzim roku 2008 byl uveden HTC Dream – první mobilní telefon používající operační systém Android. V roce 2010 se Android stal nejrozšířenějším mobilním operačním systémem.

3.2.2 Verze OS Android

Od prvního vydání testovací aplha verze OS Android v roce 2007 již uplynula na poměry dynamického odvětví mobilních operačních systémů relativně dlouhá doba, za kterou bylo vydáno mnoho verzí OS Android s postupnými vylepšeními a přidáváním nových schopností operačního systému.

První komerčně dostupná verze 1.0 byla vydána na podzim roku 2008.

Od verze 1.5 a výše se společnost Google Inc. drží pravidla, že nové verze, které přinášejí významná vylepšení jsou označována kódovým jménem nějaké

cukrovinky nebo sladkosti. Přičemž je dodržováno pravidlo, že kódové jméno každé další verze začíná vždy dalším písmenem v abecedě.

Každá nová verze OS Android typicky podporuje další novou verzi vývojářského rozhraní (API) pro tvorbu aplikací třetích stran. Novější verze API jsou zpětně kompatibilní. Navíc společnost Google Inc. vydává pro vývojáře mobilních aplikací tzv. podpůrné knihovny (Support Library), které umožňují tvůrcům aplikací využít některé funkce dostupné až v novějších verzích OS Android i na starších telefonech, pro které tyto nové verze OS Android nejsou dostupné.

Všechny oficiálně vydané verze OS Android jsou uvedeny v tabulce 3.1.

OS Android 1.0 a 1.1

Tyto verze ještě nenesly žádné kódové označení. Verze 1.0 byla vydána na podzim roku 2008 a verze 1.1 byla vydána v únoru roku 2009. První verze přinesla mimo jiné tyto funkce:

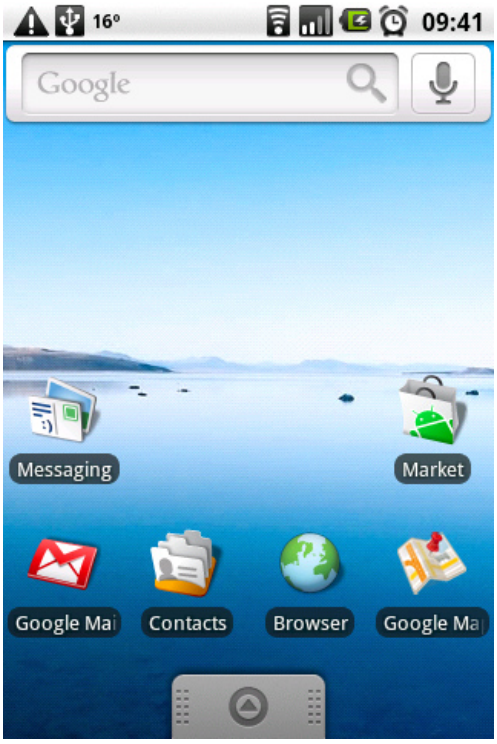
- Internetový obchod Android Market (dnešní Google Play), který umožňoval stahování a updatování aplikací třetích stran.
- Vestavěný internetový prohlížeč s plnou podporou tehdejšího standardu HTML A XHTML.
- Podporu technologií Wi-Fi a Bluetooth.
- Podporu fotoaparátu.
- Hudební a video přehrávač.
- Vestavěný YouTube video přehrávač.
- Podporu emailových protokolů POP3, IMAP4 a SMTP.

OS Android 1.5 Cupcake

Android 1.5 je první verzí, která nese kódové označení. Tato verze byla vydána v dubnu 2009, a přinesla některá vylepšení. Mezi nejvíce významné patří možnost umístění Widgetů na domácí plochu telefonu a možnost nahrávání a následného přehrávání MPEG-4 a 3GP videa.

OS Android 1.6 Donut

Tato verze byla vydána v září 2009. Tato verze přinesla mnohá vylepšení uživatelského rozhraní a dostupnosti. Například umožnila vývojářům aplikací třetích stran zahrnout jejich výsledky do vyhledávání v telefonu, zlepšila podporu pro nevidomé uživatele možností nahlas přečíst libovolný text zobrazený na displeji, přinesla podporu displejů s rozlišením WVGA (768 x 480 obrazových bodů) a vylepšení uživatelských gest.



Obrázek 3.15: Android 1.6 Donut uživatelské rozhraní(Zdroj: (46))

OS Android 2.0 a 2.1 Eclair

Android 2.0 byl vydán v říjnu roku 2009, verze 2.1 byla vydána v prosinci roku 2009. Tyto verze přinesly podporu standardu HTML5 a Bluetooth 2.1. Zároveň byla přidána podpora displejů s vyšším rozlišením a byly vylepšeny mnohé prvky uživatelského rozhraní na základě zpětné vazby uživatelů.

OS Android 2.2 Froyo

Tato verze byla vydána v květnu roku 2010. Ve verzi 2.2 bylo vývojáři přidáno mnoho nových technologií a s nimi související vylepšení chování OS. Mezi nejvýznamnější změny patří:

- Podpora vzdálených PUSH notifikací pomocí služby C2DM (Android Cloud to Device Messaging), která je integrována v Google Play services.
- Zvýšení rychlosti běhu aplikací vytvořených v programovacím jazyku Java pomocí tzv. JIT(Just-in-time) kompilace. Tedy pomocí kompilace částí bytecode za běhu přímo pro procesor daného mobilního zařízení.
- Podpora pro USB tethering.

- Možnost tvorby Wi-Fi přístupového bodu (hotspot) pro ostatní zařízení v okolí přímo z telefonu.
- Možnost vypnutí mobilních dat.
- Podpora technologie Adobe Flash.

OS Android 2.3 Gingerbread

Android 2.3 byl vydán v prosinci roku 2010. Postupně byly vydávány drobné opravy až do verze 2.3.7, která byla vydána v září 2011. Verze OS Android 2.3 přidala podporu některých nových technologií, mezi nejdůležitější patří:

- Podpora technologie NFC (Near Field Communication).
- Nativní podporu voIP telefonie.
- Podpora WebM/VP8 video kodeku, který je používán mimo jiné v HTML5.
- Rozšířena podpora možnosti kopírování a následného vložení textu.
- Pro zvýšení výkonu mobilních aplikace přidána možnost paralelního spuštění garbage collection při běhu aplikace.

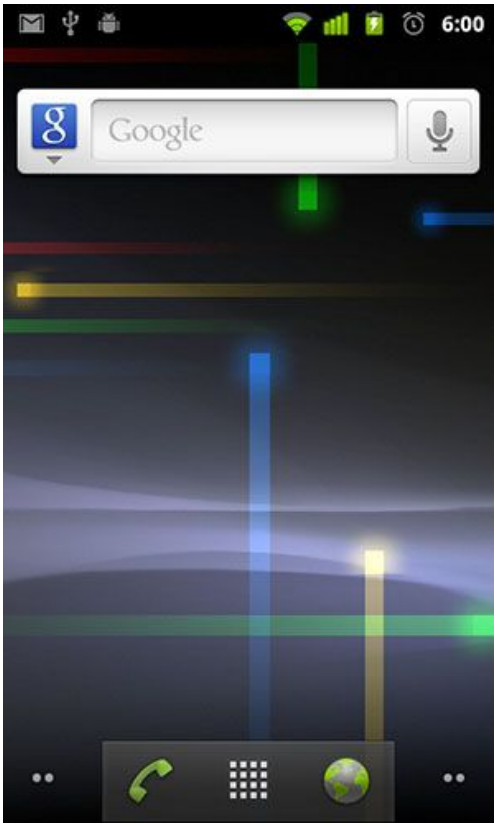
OS Android 3.1, 3.2 a 3.3 Honeycomb

Android 3.0 byl vydán v únoru roku 2011. Poslední update na verzi 3.2.6 byl vydán až v únoru roku 2012.

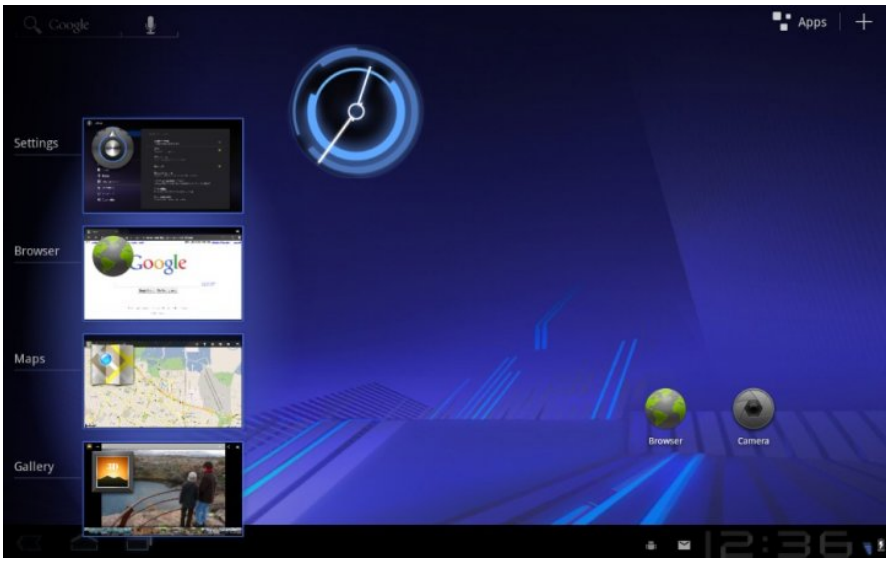
V roce 2010 došlo k prudkému rozvoji trhu s tablety a společnost Google Inc. na toto musela velmi rychle reagovat. Výsledkem bylo vydání právě verzí OS Android 3.x, které byly určeny pouze pro tablety a nikoliv mobilní telefony. Tato verze tedy přinesla mnoho změn a vylepšení uživatelského rozhraní, které vylepšovaly chování OS Android na displejích větších úhlopříček, které tablety mají. Kromě změn uživatelského rozhraní byla ještě přidána podpora vícejádrových procesorů a možnost připojení externích klávesnic, případně myši.

OS Android 4.0 Ice Cream Sandwich

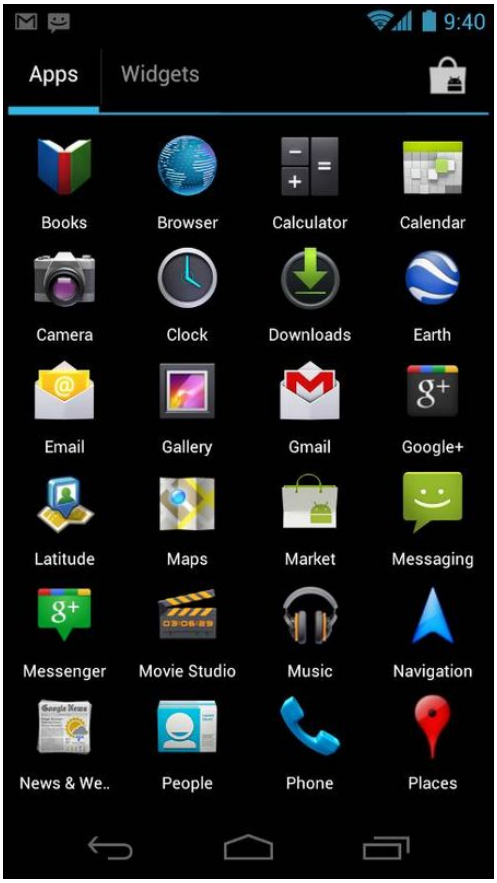
Android verze 4.0 byl vydán v říjnu roku 2011. Poslední update na verzi 4.0.4 byl vydán v březnu roku 2012. Tato verze přinesla mnoho vylepšení uživatelského rozhraní a přidala nové vestavěné aplikace, jako například editor obrázků. Mimo jiné přinesla také možnost hardwarové akcelerace uživatelského rozhraní a možnost nahrávání 1080p videa.



Obrázek 3.16: Android 2.3 Gingerbread uživatelské rozhraní(Zdroj: (46))



Obrázek 3.17: Android 3.2 Honeycomb uživatelské rozhraní(Zdroj: (46))



Obrázek 3.18: Android 4.0 Ice Cream Sandwich(Zdroj: (46))

OS Android 4.1, 4.2 a 4.3 Jelly Bean

Android verze 4.1 byl vydán v červenci roku 2012. Poslední update na verzi 4.3.1 byl vydán v říjnu roku 2013.

Ve verzi 4.1 se vývojáři zaměřili hlavně na plynulost uživatelského rozhraní. Způsob zobrazování byl upraven tak, aby obraz byl zobrazován vždy rychlostí 60 snímků za sekundu a byla přidána vertikální synchronizace obrazu i mezi jednotlivými aplikacemi. V této verzi byla také odstraněna nativní podpora technologie Adobe Flash.

Ve verzi 4.3 byla přidána podpora Bluetooth LE (low energy), podpora displejů až do rozlišení 4K, Open GL ES3.0 pro rozšíření možností grafických efektů ve hrách, a byla vylepšena správa zařízení, kterým je povoleno připojení přes USB v debug režimu.

OS Android 4.4 Kitkat

Android verze 4.4 byl vydán v říjnu roku 2013. Poslední update na verzi 4.4.4 byl vydán v červnu roku 2014.

V této verzi bylo přidáno například vývojářské API pro infračervené dálkové ovladače a podporu tisku přes Wi-Fi. Navíc je zde obsaženo tzv. Android Runtime, nové prostředí pro spouštění aplikací třetích stran, které nahrazuje stávající virtuální stroj Dalvik. V této verzi byl v základu stále použit virtuální stroj Dalvik a Android Runtime bylo možné zapnout pouze po dodatečném zásahu.

OS Android 5.0 a 5.1 Lolipop

Android verze 5.0 byl vydán v listopadu roku 2014. Poslední update na verzi 5.1.1 byl vydán v dubnu roku 2015. Tato verze OS Android přinesla podporu mnoha nových funkcí, nejdůležitější z nich jsou:

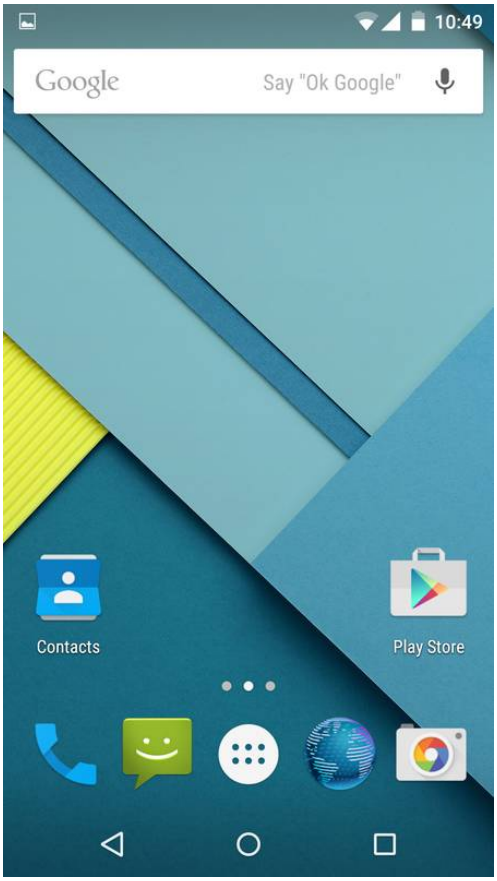
- Finální nahrazení virtuálního stroje Dalvik novým Android Runtime, které provádí předkompilování aplikace před prvním spuštěním.
- Podpora 64bitových procesorů.
- Material design, nový přepracovaný vzhled uživatelského rozhraní.
- Optimalizace spotřeby energie, které vyplynuly z tzv. Projektu Volta.
- Pro zvýšení výkonu mobilních aplikací přidána možnost paralelního spuštění garbage collection při běhu aplikace.

OS Android 6.0 Marshmallow

Android verze 6.0 byl vydán v říjnu roku 2015. Poslední update na verzi 6.0.1 byl vydán v prosinci roku 2015. Tato verze přináší podporu USB konektoru typu C, podporu 4k rozlišení pro více aplikací a možnost dodatečného zákazu některých oprávnění, která uživatel povolil aplikaci třetí strany během její instalace.

OS Android 7.0 Nougat

Android verze 7.0 je zatím poslední verzí OS Android. Tato verze byla vydána v srpnu roku 2016. V této verzi byl přidán nový způsob kompilace aplikací za běhu (JIT), který si klade za cíl zajistit o 75% rychlejší instalaci aplikací a o polovinu menší velikost předkompilovaného kódu. Dále tato verze přináší drobná vylepšení uživatelského rozhraní.



Obrázek 3.19: Android 5.1 Lolipop uživatelské rozhraní(Zdroj: (46))

3.2.3 Podíl verzí OS Android na trhu

Společnost Google Inc. vede statistiku verzí operačního systému Android, které přistupují do internetového obchodu Google Play. Do obchodu Google Play mohou přistupovat mobilní zařízení s OS Android 2.2 a vyšším. Avšak podíl starších verzí OS na trhu je naprosto okrajový, neboť bez možnosti použití obchodu Google Play ztrácí telefon velkou část svého potenciálu, a tato zařízení již nejsou ve větší míře běžně používána.

V tabulce 3.2 je uvedeno procentuální zastoupení jednotlivých verzí OS Android na trhu během časového období mezi 29. srpnem 2016 a 5. zářím 2016, toto zastoupení je určeno dle přístupů do obchodu Google Play. V této tabulce nejsou zahrnuty podíly verzí OS, které jsou nižší, než 0.1%.

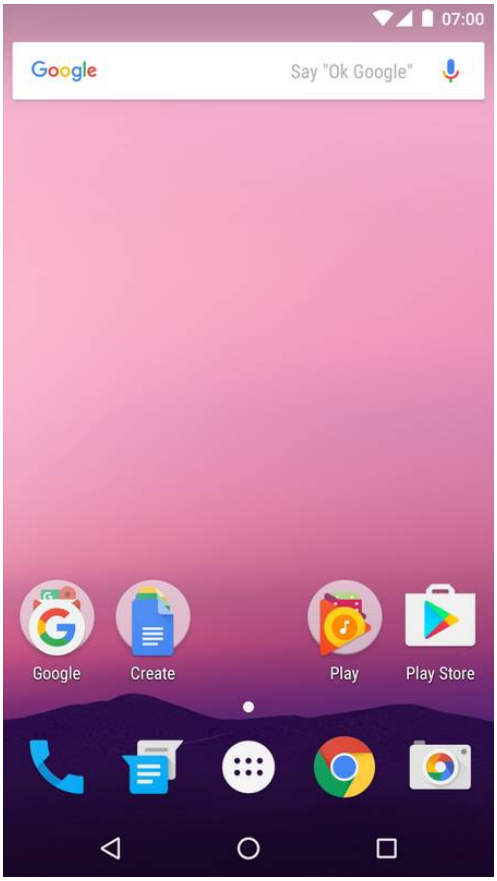
Z této tabulky vyplývá, že naprostá většina uživatelů používá verzi OS Android 4.0 a vyšší. Z tohoto důvodu je v dnešních dnech také velká část nově vytvářených mobilních aplikací cílena na mobilní zařízení s OS Android 4.0 a novější (verze vývojářského API 14 a vyšší).

Tabulka 3.1: Verze OS Andoid

Kódové označení	Číslo verze	Datum vydání	Verze vývojářského API
–	1.0	23. září 2008	1
–	1.1	9. února 2009	2
Cupcake	1.5	27. dubna 2009	3
Donut	1.6	15. září 2009	4
Eclair	2.0 - 2.1	26. října 2009	5 - 7
Froyo	2.2 - 2.2.3	20. května 2010	8
Gingerbread	2.3 - 2.3.7	6. prosince 2010	9 - 10
Honeycomb	3.0 - 3.2.6	22. února 2011	11 - 13
Ice Cream Sandwich	4.0 - 4.0.4	18. října 2011	14 - 15
Jelly Bean	4.1 - 4.3.1	9. července 2012	16 - 18
KitKat	4.4 - 4.4.4	31. října 2013	19 - 20
Lollipop	5.0 - 5.1.1	12. listopadu 2014	21 - 22
Marshmallow	6.0 - 6.0.1	5. října 2015	23
Nougat	7.0 - 7.1	22. srpna 2016	24 - 25

Tabulka 3.2: Podíl verzí OS Android na trhu k 5. září 2016 (Zdroj: (15))

Kódové označení	Číslo verze	Podíl na trhu
Froyo	2.2	0.1%
Gingerbread	2.3.3 - 2.3.7	1.5%
Ice Cream Sandwich	4.0.3 - 4.0.4	1.4%
Jelly Bean	4.1.x	5.6%
Jelly Bean	4.2.x	7.7%
Jelly Bean	4.3	2.3%
KitKat	4.4	27.7%
Lollipop	5.0	13.1%
Lollipop	5.1	21.9%
Marshmallow	6.0	18.7%



Obrázek 3.20: Android 7.0 Nougat uživatelské rozhraní(Zdroj: (46))

3.2.4 Architektura OS Android

Architektura OS Android se skládá z vrstev, které spolu vzájemně spolupracují. Graficky je architektura znázorněna na obrázku 3.21. Konkrétně se architektura OS Android skládá z následujících vrstev:

Jádro

Android využívá jádro operačního systému Linux ve verzi 2.6 a vyšší. Konkrétní verze použitého jádra závisí na verzi OS Android a na hardwaru daného mobilního zařízení, pro který je OS Android kompilován. Jádro obsahuje ovladače, které zajišťují komunikaci s hardwarem konkrétního zařízení. Jádro obstarává dále správu procesů, paměti, síťového spojení či napájení.

Z Linuxového jádra byly odstraněny knihovny GNU C (glibc), které jsou použity u většiny Linuxových distribucí. Dále byl odstraněn X server (linuxové distribuce typicky využívají Xorg) a tudíž není na platformě Android možné

spustit grafické aplikace určené pro Linux(24). Jádro je napsáno v jazyce C a C++.

Knihovny

Knihovny poskytující základní funkcionalitu operačního systému jsou napsány v jazyce C nebo C++. Konkrétně se jedná o knihovny pro zobrazování aplikací a jejich vrstvení (Surface Manager), práci s grafikou (Open GL, SGL), práci s daty (SQLite) či multimédii (knihovna Media). Mezi knihovnami je také obsaženo vykreslovací jádro WebKit pro internetový prohlížeč a knihovna SSL, která se stará o šifrování a zabezpečený přenos dat v síti.

Android Runtime a Dalvik Virtual Machine

Tato vrstva slouží pro spouštění aplikací. Aplikace, které jsou psány v Javě, jsou nejdříve převedeny na Java bytecode a následně na tzv. Dalvik bytecode pro pro Dalvik virtuální stroj. Dalvik bytecode je uložen v .dex souboru, který je obsažen v .apk instalačních archivech aplikací. Pro samotné spuštění aplikace se do verze OS Android 5.0 používá Dalvik Virtual Machine, který používá just-in-time(JIT) kompilaci.

Od verze OS Android 5.0 a výše je použito tzv. Android Runtime, které bylo pro vývojáře dostupné již ve verzi OS Android 4.4. Android Runtime využívá tzv. ahead-of-time kompilaci(AOT). Převod aplikace do nativního kódu zařízení proběhne vždy pouze jednou po instalaci aplikace. Výsledný binární kód zůstane uložen v mobilním zařízení spolu s instalačním .apk archivem. Díky této úpravě byla snížena náročnost běhu aplikací a tím zvýšena výdrž zařízení na jedno nabití baterie.

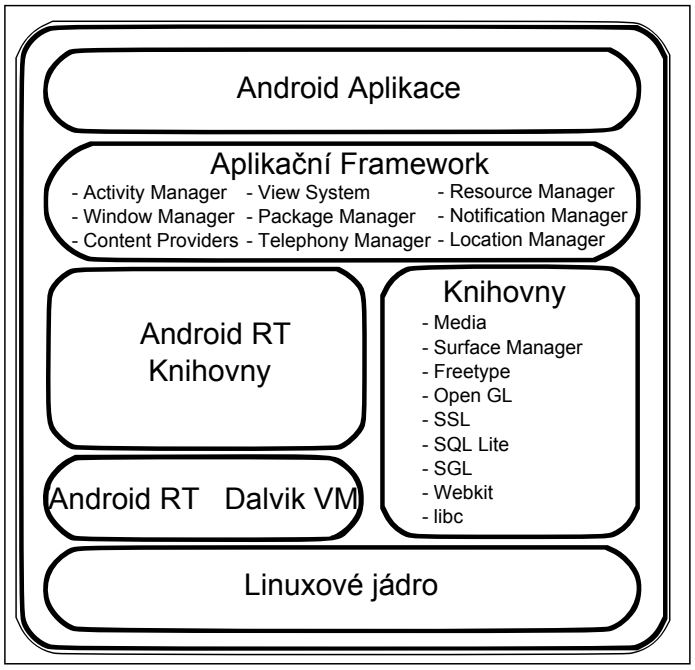
Application Framework

Vrstva tvořící systémové rozhraní obsahující knihovny napsané v Javě. Tato vrstva je nejdůležitější pro vývojáře mobilních aplikací, neboť používají její rozhraní pro přístup k jednotlivým službám, které OS Android poskytuje pro aplikace třetích stran.

Konkrétně tato vrstva zajišťuje správu životního cyklu aplikací (Activity Manager), umožňuje zjištění polohy (Location Manager), správu balíčků aplikací (Package Manager), zjištění informací o stavu telefonu, jako informace o síti, zda probíhá hovor apod.(Telephony Manager). Dále je v této vrstvě obsažena správa notifikací(Notification Manager) a správa sdílení informací mezi aplikacemi(Content Providers).

Android aplikace

Android aplikace jsou nejvyšší vrstvou celé architektury. Do této vrstvy se řadí jak předinstalované aplikace, tak aplikace třetích stran.



Obrázek 3.21: Schéma vrstev OS Android (Zdroj: (27))

3.3 Možnosti vývoje aplikací pro OS Android

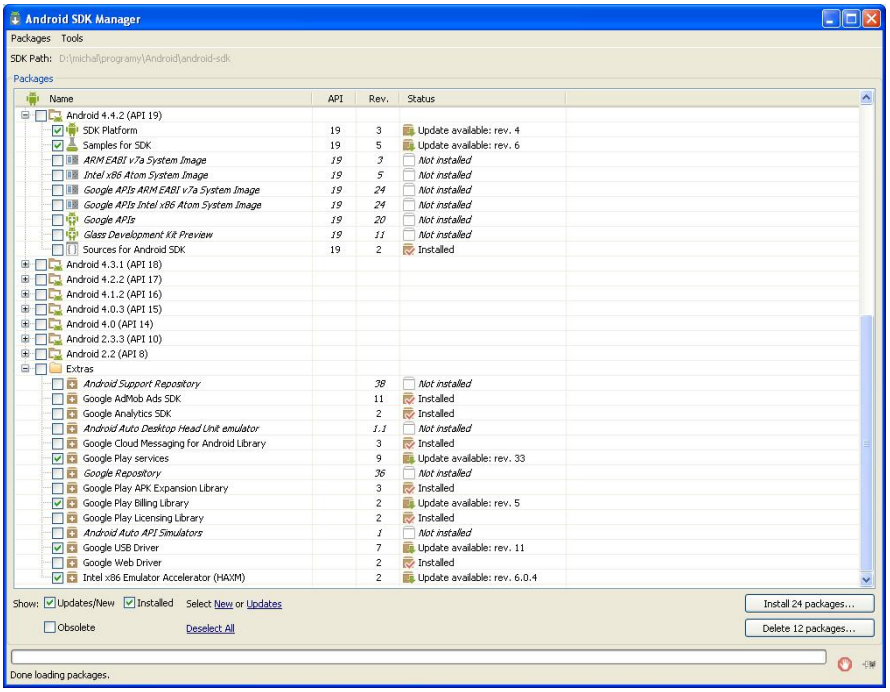
Jádro a knihovny OS Android jsou naprogramovány v jazyce C a C++. Pro OS Android je sice také možné vytvářet aplikace třetích stran v jazyce C a C++ s využitím vývojářské sady nástrojů Native Development Kit (NDK), které poskytuje společnost Google Inc. zdarma vývojářům. Ovšem vývoj v jazyce C a C++ pro naprostou většinu aplikací nepřináší žádné výhody oproti vývoji v jazyce Java. Naopak přináší mnohé komplikace s náročnější správou paměti a celkovou komplexností kódu. Proto vývoj v jazyce C a C++ s výjimkou výpočetně a graficky velmi náročných aplikací není doporučován(19).

Pro tvorbu aplikací třetích stran je typicky využíván programovací jazyk Java. Pro tvorbu aplikací v tomto jazyku je společností Google Inc poskytnuta sada vývojářských nástrojů Android SDK Tools, kterou je třeba použít s některým z podporovaných vývojových prostředí.

3.3.1 Android SDK Tools

Sada nástrojů Android SDK Tools obsahuje správce balíčků Android SDK Manager, který umožňuje stahovat a instalovat různé balíčky a nástroje potřebné k vývoji pro různé verze OS Android.

Další významnou součástí Android SDK Tools je Android Virtual Devices (AVD) Manager který slouží ke správě virtuálních zařízení(emulátorů), díky



Obrázek 3.22: Android SDK Manager (Zdroj: Autor)

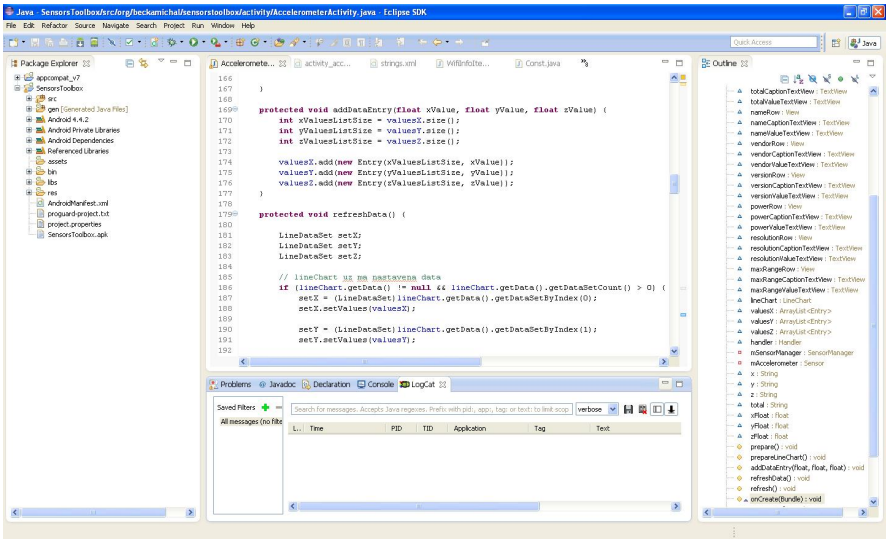
kterým je možné otestovat vytvářenou aplikaci pro různá rozlišení displeje a konfigurace mobilních telefonů. Bohužel i když bylo pro Android emulátor vydáno již mnoho optimalizací za účelem zrychlení jeho běhu, je na starších počítačích pro vývoj stále nevhodný kvůli své velmi nízké rychlosti.

Android SDK Manager umožňuje stáhnout a doinstalovat USB ovladače pro spouštění aplikace přes USB kabel na reálném zařízení. Dále je zde možnost stáhnout různé podpůrné knihovny, například Android Support library, Google Play services, Google Analytics SDK nebo Google AdMob Ads SDK umožňující umístění placené reklamy v aplikaci.

3.3.2 Vývojová prostředí

Do konce roku 2015 bylo oficiálně podporovaným vývojovým prostředím pro OS Android prostředí Eclipse, v kterém je po instalaci ADT pluginu možné vyvíjet, testovat a vytvářet buildy aplikací pro OS Android(14). Na přelomu roku 2015 a 2016 ovšem došlo k ukončení oficiální podpory ze strany společnosti Google Inc., která se jako nové oficiální vývojové prostředí rozhodla použít nově vytvářený nástroj Android Studio, který je založen na vývojovém prostředí IntelliJ IDEA.

Oproti prostředí Eclipse nevyžaduje Android Studio instalaci a nastavení ADT pluginu, a jeho instalace a zprovoznění je obecně uživatelsky přívětivější.



Obrázek 3.23: Vývojové prostřední Eclipse(Zdroj: Autor)

Velkou změnou v novém vývojovém prostředí Android Studio je změna buildovacího nástroje. Zatímco Eclipse využívá pro buildování Android aplikací nástroj Ant, Android Studio využívá nově Gradle. Gradle umožňuje nově linkování buildu aplikace přímo s knihovnami, které se dynamicky stahují z online repozitářů bez nutnosti ručního přilinkování .jar balíčků.

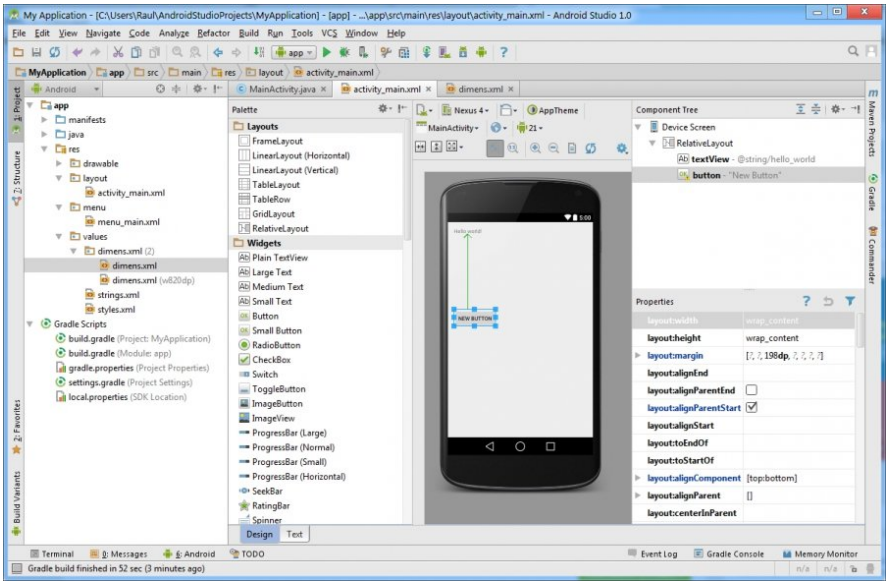
Velkou nevýhodou buildovacího nástroje Gradle je ovšem rychlost. Při opakovaném měření rychlosti buildování na rozsáhlé referenční aplikaci, pro kterou vytvořil nástroj Ant build za 20 sekund bylo zjištěno, že pro stejnou aplikaci na stejném hardware vytváří Gradle build 225 sekund. Doba tvorby buildu aplikace může tedy s nástrojem Gradle být u některých aplikací i 11 krát delší, než při použití nástroje Ant. Měření bylo provedeno s nástrojem Gradle integrovaným v Android Studio 2.0.0. Referenční aplikace obsahovala více, než 64 tisíc metod, byl tedy při buildu použit multi-dex pro rozdělení výsledného .dex souboru (Zdroj měření: Autor).

Společnost Google Inc. je si tohoto problému vědoma a postupně vydává optimalizace, které si kladou za cíl zrychlení tvorby buildů pomocí nástroje Gradle. Ovšem i přes tuto snahu je tvorba buildů s nástrojem Gradle stále řádově pomalejší, než se starším nástrojem Ant.

Tento problém se projevuje nejvíce u velmi rozsáhlých aplikací, nebo při vývoji i malých aplikací na PC, které je vybaveno slabším hardwarem.

3.4 Základní součásti Android aplikace

Tato část práce charakterizuje základní součásti, z kterých se skládá Android aplikace. Konkrétní aplikace nemusí nutně využívat všechny zde uvedené zák-



Obrázek 3.24: Vývojové prostřední Android Studio(Zdroj: (20))

ladní části aplikace. Ovšem ty, které používá, je nutné uvést v souboru AndroidManifest.xml, který je uložen v kořenovém adresáři projektu dané aplikace.

3.4.1 AndroidManifest

AndroidManifest je xml soubor, který je umístěn v kořenovém adresáři aplikace.

Tento soubor zároveň slouží, jako informace pro OS Android, aby věděl, o jakých událostech má aplikaci informovat, nebo jaké informace dokáže tato aplikace předat jiným aplikacím nainstalovaným v daném mobilním zařízení.

Zároveň AndroidManifest obsahuje informaci o minimální podporované a o cílové verzi vývojářského API. Jsou zde vyjmenována všechna oprávnění, která aplikace vyžaduje pro svůj běh. V souboru AndroidManifest je také uveden název aplikace a cesta k ikoně aplikace.

3.4.2 Activity

Activity je základním prvkem uživatelského rozhraní aplikace. Activity reprezentuje typicky (v naprosté většině případů) jednu obrazovku aplikace. Android aplikace, kterou je možné otevřít a zobrazit musí mít alespoň jednu Activity, která je definována, jako tzv. parent activity, která se zobrazí po startu aplikace. Typicky aplikace obsahuje více Activities, mezi kterými uživatel v aplikaci přechází podobně, jako přechází mezi jednotlivými webovými stránkami v internetovém prohlížeči.

Activity se typicky skládá ze dvou částí. Z logiky, která je implementována v jazyce Java, a z grafického layoutu(rozložení) obrazovky, které je definováno v přidruženém xml souboru.

Při vytvoření Activity systém vytvoří nový proces a do operační paměti alokuje data a elementy uživatelského rozhraní. Protože při postupném průchodu uživatele hierarchií Activities jsou vytvářeny stále další Activity, bylo by velmi náročné udržovat v běhu vytvořené procesy pro Activity, které nejsou momentálně zobrazeny. Proto Application Framework obsahuje správce Activity Manager, který řídí přechody mezi Activities. Konkrétně tak, že do zásobníku přidává informace o navštívených Activity, nebo případně tyto informace odebírá, pokud se uživatel vrátí o jednu Activity zpět. Pro řízení životního cyklu a stavu, ve kterém se daná konkrétní Activity právě nachází používá Activity Manager stav Activity. Stavy Activity a přechody mezi nimi jsou grafiky znázorněny v obrázku 3.25. Konkrétně se Activity může nacházet v následujících stavech:

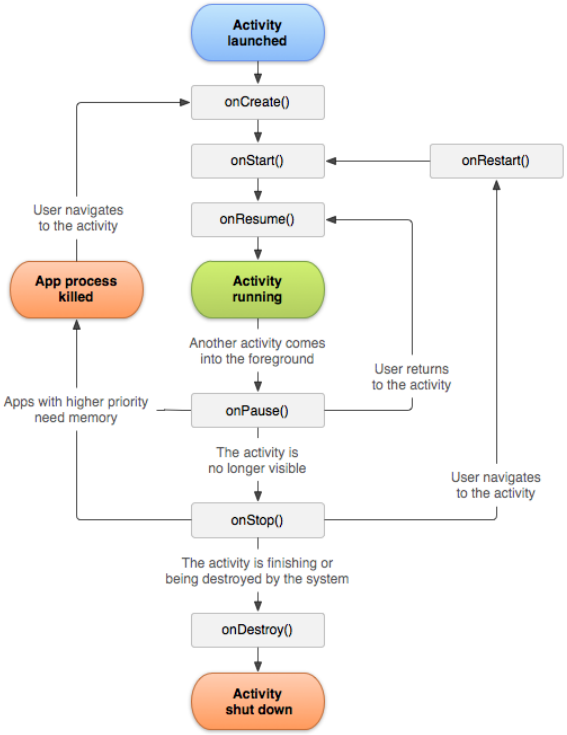
- Created state – Ve stavu Created byla Activity inicializována, byl načten layout(rozložení) Activity a alokována většina datových struktur do operační paměti. Activity není v tomto stavu viditelná uživateli. V tomto stavu se Activity nenachází po delší dobu a okamžitě po alokování všech potřebných struktur přechází do následujícího stavu Started state.

- Started state – Jakmile Activity přechází do stavu Started, je zobrazena uživateli. V tomto stavu se Activity nachází také pouze po velmi krátkou dobu a následně automaticky přechází do následujícího stavu Resumed state.

- Resumed state – V tomto stavu je Activity na popředí a viditelná uživateli, který může používat uživatelské rozhraní Activity (uživatel může například stisknout tlačítka nebo psát text). V tomto stavu se v aplikaci může nacházet maximálně jedna Activity, se kterou uživatel právě smí interagovat. V tomto stavu má tato Activity prioritu při přidělování výpočetního výkonu před ostatními Activity, aby byla zajištěna plynulost uživatelského rozhraní aplikace.

- Paused state – Do tohoto stavu se Activity dostane pokud je částečně překryta jinou Activity, ale její layout(rozložení) je stále viditelné (například je překryta jen částečně, nebo překrývající aktivita používá aplha průhlednost). V tomto stavu Activity nemůže ovšem interagovat s uživatelem.

- Stopped state – Ve stavu Stopped není Activity viditelná uživateli (je na pozadí). Typicky se do tohoto stavu Activity dostane, pokud je překryta jinou Activity. V tomto stavu má Activity alokována stále všechna data v operační paměti, ale nemůže spouštět žádný kód. Je mnohem šetrnější

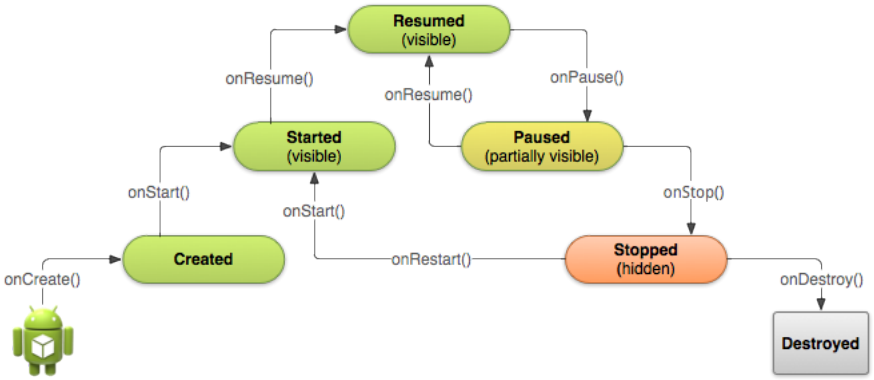


Obrázek 3.25: Životní cyklus Activity(Zdroj: (26))

udržovat Activity v tomto stavu pro případ, že by se uživatel k této Activity chtěl vrátit, než znovu vytvářet a alokovat Activity znovu. Pokud se uživatel k této Activity znovu vrátí, přechází do Started state.

- Destroyed state – Do tohoto stavu se Activity dostane, pokud Activity Manager rozhodne, že není třeba tuto Activity dále mít alokovanou v operační paměti. Může k tomu dojít z důvodu, že uživatel se z této Activity vrátí zpět na předchozí, nebo také kvůli nedostatku operační paměti telefonu.

Z obrázku 3.25 je také patrné, že každý callback při přechodu mezi stavy má svůj protějšek. Struktury, které byly alokovány v metodě onCreate() by měly být dealokovány v metodě onDestroy(). Úlohy, které byly spuštěny v metodě onStart() by měly být zastaveny v metodě onStop(). Stejně tak činnosti, které byly započaty v metodě onResume() by měly být ukončeny v metodě onPause().



Obrázek 3.26: Stavy Activity(Zdroj: (16))

3.4.3 Service

Service je procesem, který běží na pozadí i pokud uživatel otevře jinou aplikaci. Service se používá typicky pro déle trvající úlohy, které běží i bez nutnosti grafického rozhraní pro uživatele. Například stahování dat na pozadí, pokud uživatel používá jinou aplikaci, nebo přehrávání hudby.

Pro řízení životního cyklu Service jsou, podobně jako u Activity, používány stavy. Service se může nacházet konkrétně ve třech stavech:

- Starting state – Service je vytvářen, je volán callback onCreate().
- Running state – Service je spuštěn a běží na pozadí. Je volán callback onStartCommand(), případně onBind() pokud se jedná o tzv. Bound service.
- Destroyed state – Service je ukončen. Service může být ukončen jak svou vlastní činností, tak aktivitou uživatele.

Service je možné spustit pomocí dvou odlišných způsobů. Každý z nich má svá specifika a hodí se pro jiné typy úloh. Konkrétně se jedná o tyto způsoby spuštění Service:

- Spuštění pomocí volání startService(). Takto vytvořený proces typicky nevrací svůj výsledek objektu, který jej vytvořil a po dokončení své činnosti se ukončí sám. Typicky je jedná například o stahování souboru na pozadí.
- Spuštění pomocí volání bindService() sváže Service s objektem, který jej spustil. S takto vytvořeným procesem je možné komunikovat a získávat informace o jeho průběhu. K jednomu Service se může navázat více objektů (komponent) pomocí volání bindService(), a všechny mohou s Ser-

vice komunikovat. Service je ukončen ve chvíli, kdy se od něj pomocí volání unbindService() odpojí všechny svázané objekty(komponenty).

3.4.4 Broadcast Receiver

Broadcast Receiver slouží k naslouchání změn nějakého druhu událostí v telefonu. Události jsou typicky vysílány systémem (ale mohou být vysílány i jinou aplikací). Jedná se například o informace o tom, že na telefon je příchozí hovor, že byl vypnut nebo zapnut displej telefonu, že se změnila GPS poloha telefonu nebo došlo ke změně stavu baterie.

V souboru AndroidManifest.xml je možné deklarovat, jakým událostem bude aplikace naslouchat, a jakému objektu aplikace má být případná příchozí událost předána, aby se mohl příslušně zachovat. Aplikace může být schopná přijímat více typů událostí. Toto je možné využít například pro aktualizaci polohy uživatele na mapě při změně GPS polohy. Případně dokonce pro přepnutí příslušné Activity na popředí, nebo zobrazení notifikace uživateli, pokud aplikace obdrží určitou událost.

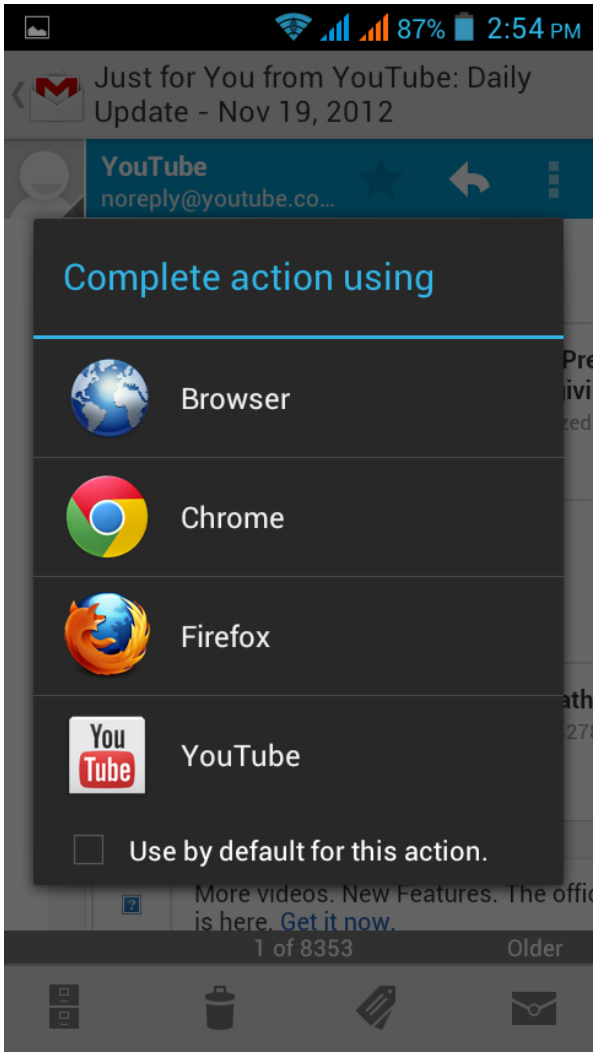
3.4.5 Intent

Intent je označením pro zprávy, které je možné zasílat jak mezi jednotlivými částmi aplikace (typicky mezi Activity), tak mezi mezi aplikací a systémem. Existují dva druhy Intent. Konkrétně explicitní, které mají přesně určeného příjemce (například konkrétní Activity). A implicitní, kde není přesně určen konkrétní příjemce, pouze je popsán druh zamýšlené akce (například otevřít internetovou stránku, nebo přehrát MP3 soubor). Jakmile systém obdrží implicitní Intent, a zjistí, že v systému je pro tento typ akce zaregistrováno více aplikací, nabídne uživateli dialog pro výběr, kterou aplikaci má systém použít pro zpracování daného Intent. Příklad dialogu pro otevření URL adresy internetové stránky je uveden v obrázku 3.27.

3.4.6 Content Provider

Content Provider je deklarace, která slouží ke sdílení dat mezi jednotlivými aplikacemi, které jsou nainstalovány v mobilním zařízení. Pomocí Content Provider je možné dekrarovat rozhraní (API), pomocí kterého ostatní aplikace mohou přistupovat k datům, které jim tato aplikace pomocí tohoto rozhraní poskytne.

Typicky je Content Provider používán pro přístup k seznamu telefonních kontaktů, k SMS zprávám uloženým v telefonu nebo k nastavení telefonu. Podobné rozhraní může ovšem také deklarovat libovolná aplikace třetích stran pro přístup k datům, která poskytuje.



Obrázek 3.27: Dialog pro dokončení implicitní Intent(Zdroj: (40))

Obrázek 3.28: Použití Fragmentu(Zdroj: (21))

3.4.7 Fragment

Fragmenty nepatří mezi základní složky aplikace. S příchodem tabletů byli vývojáři OS Android nuceni vytvořit novou komponentu, která by umožňovala modulárně zobrazovat části uživatelského rozhraní aplikace tak, aby bylo možné celkové rozložení uživatelského rozhraní přizpůsobit velikosti displeje mobilního zařízení. K tomuto účelu byly vytvořeny Fragmenty.

Fragment reprezentuje část uživatelského rozhraní v rámci Activity. Jedna Activity může zobrazovat více Fragmentů, a jeden Fragment může být ve více instancích zobrazen ve více Activity. Díky tomuto konceptu je možné upravit uživatelské rozhraní aplikací tak, že na tabletu je v jeden okamžik vedle sebe možné zobrazit dvě, či více obrazovek aplikace, které jsou na mobilním telefonu zobrazeny samostatně. Fragmenty byly do OS Android přidány od verze 3.0 a výše (verze vývojářského API 11).

Fragment má svůj životní cyklus, který je ovšem podřízen životnímu cyklu Activity, jejíž je součástí. Pokud je Activity pozastavena, nebo ukončena, Fragment je převeden do stejného stavu. Fragment může být k dané Activity připojen pomocí tzv. FragmentTransaction, která slouží k zajištění toho, aby případná úprava více Fragmentů v jedné Activity byla provedena transakčně. Mnohem častěji používaným způsobem připojení Fragmentu k Activity je přímo v xml souboru, který obsahuje layout(rozložení) Activity.

Životní cyklus Fragmentu je velmi podobný životnímu cyklu Activity. Propojení životního cyklu Activity a Fragmentu je znázorněno na obrázku 3.29. Rozdílem je přidání několika callback volání, které informují fragment o změnách stavu, které souvisí s připojováním a odpojováním od Activity. Konkrétně se jedná o následující callback volání:

- onAttach() – Metoda onAttach() je volána v momentě, kdy je Fragment připojen k Activity.
- onCreateView() – Tato metoda je volána ve chvíli, kdy je vytvořen layout(rozložení) Fragmentu. Metoda onCreateView() slouží k vytvoření

grafických objektů, které náleží Fragmentu, a případných referencí na ně v kódu.

- onActivityCreated() – Tato metoda je volána ve chvíli, kdy proběhne volání metody onCreate() Activity, ke které je Fragment připojen.
- onDestroyView() – Tato metoda je volána ve chvíli, kdy je layout(rozložení) Fragmentu odstraněno z layoutu(rozložení) Activity.
- onDetach() – Metoda onDetach() je volána ve chvíli, kdy je Fragment odpojen od Activity.

3.5 Struktura projektu v prostředí Eclipse

Tato část charakterizuje strukturu projektu Android aplikace ve vývojovém prostředí Eclipse. Toto vývojové prostředí bylo použito pro vývoj aplikace v praktické části práce.

Adresářová struktura projektu obsahuje následující významné adresáře:

3.5.1 Adresář src

Zkratka „src“ je zkrácením slova „sources“, kterým jsou myšleny zdrojové kódy aplikace v jazyce Java. Tyto zdrojové kódy popisují samotné chování aplikace.

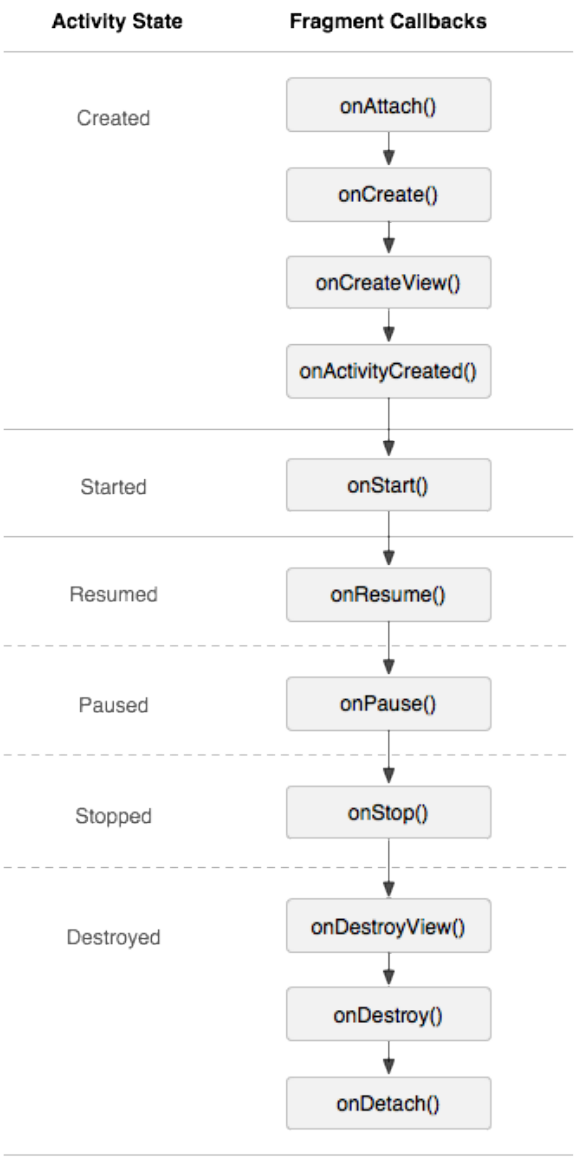
Při pojmenování balíčků a tříd je nepsaným pravidlem používat jako název balíčku reverzní doménu, která jednoznačně identifikuje, vývojáře aplikace a následně samotnou aplikaci. Například jako reverzní doména v praktické části práce byla použita doména „org.beckamichal.sensorstoolbox“, kde „org“ je identifikátorem domény nejvyššího řádu. Konkrétně doména .org dříve sloužila pro potřeby neziskových organizací, a dnes je hojně využívána tvůrci otevřeného a neziskového softwaru.

Druhá část reverzní domény „beckamichal“ je identifikátorem vývojáře aplikace.

Třetí část reverzní domény „sensorstoolbox“ je identifikátorem konkrétní aplikace v rámci aplikací jednoho vývojáře.

3.5.2 Adresář gen

Zkratka „gen“ je zkrácením slova „generated“. Obsah tohoto adresáře je automaticky vytvářen vývojovým prostředím. Je zde obsažen automaticky generovaný soubor „R.java“, který obsahuje vygenerovaný seznam všech objektů, které jsou uloženy v adresáři „res“ (resources). Tímto umožňuje přístup k layouts(rozložení), drawables, a dalším xml souborům, jako například strings.xml přímo ze zdrojového Java kódu aplikace.



Obrázek 3.29: Vliv Activity na životní cyklus Fragmentu(Zdroj: (17))

3.5.3 Knihovny

Další částí adresářové struktury jsou knihovny Android, knihovny na, kterých je projekt závislý (Dependencies), a odkazované knihovny (Referenced Libraries).

3.5.4 Adresář assets

Do tohoto adresáře je možné uložit další podpůrné soubory aplikace. Adresář assets umožňuje vkládat podadresáře a vytvářet tak adresářovou strukturu. Obsah adresáře assets ovšem není na rozdíl od adresáře „res“ (resources) indexován v automaticky generovaném souboru „R.java“ v adresáři „gen“. Práce s adresářem assets v aplikaci se tedy spíše blíží práci se systémem souborů.

3.5.5 Adresář bin

Zkratka „bin“ je zkrácením slova „binary“. Tento adresář obsahuje .class a další soubory připravené k tvorbě .apk archivu aplikace.

3.5.6 Adresář libs

Zkratka „libs“ je zkrácením slova „libraries“. Tento adresář obsahuje .jar archivy knihoven, které může aplikace využít.

3.5.7 Adresář res

Zkratka „res“ je zkrácením slova „resources“. Tento adresář obsahuje většinu důležitých součástí aplikace, které nejsou obsaženy přímo v Java kódu aplikace. Obsah tohoto adresáře je automaticky indexován v souboru „R.java“ v adresáři „gen“.

Konkrétně tento adresář obsahuje následující podadresáře:

Podadresáře drawable

Tyto podadresáře obsahují grafické elementy, které je možné vykreslit v uživatelském rozhraní aplikace.

Protože operační systém Android je určen pro širokou škálu mobilních zařízení s různými velikostmi a rozlišeními displeje, bylo při návrhu uživatelského rozhraní nevhodné použít, jako základní jednotku běžný pixel (px), který je běžně používán například při návrhu internetových stránek. Neboť například text s pevně danou velikostí v počtu pixelů by na některých telefonech byl příliš velký, a na jiných by se stával nečitelný kvůli přílišné jemnosti rozlišení displeje.

Z tohoto důvodu byla vytvořena jednotka dp (density independent pixel), který je možné vypočítat ze vzorce 1dp = 160px/dpi. Kde dpi je jemnost

displeje vyjadřující kolik pixelů se vejde do jednoho palce na displeji. Jednotka dp je primárně používanou jednotkou k určování velikosti elementů grafického rozhraní aplikací v operačním systému Android.

Na základě jemnosti displeje byly mobilní zařízení rozčleněny do několika skupin:

- ldpi (low density) – Zařízení s nízkou hustotou bodů na displeji, přibližně 120dpi.
- mdpi (medium density) – Zařízení se střední hustotou bodů na displeji, přibližně 160dpi.
- hdpi (high density) – Zařízení s vysokou hustotou bodů na displeji, přibližně 240dpi.
- xhdpi (extra-high density) – Zařízení s hustotou bodů na displeji, přibližně 320dpi.
- xxhdpi (extra-extra-high density) – Zařízení s hustotou bodů na displeji, přibližně 480dpi.
- xxxhdpi (extra-extra-extra-high density) – Zařízení s hustotou bodů na displeji, přibližně 640dpi.

Adresáře drawable jsou sadou několika adresářů, jejichž název začíná prefixem „drawable“, za kterým je označení hustoty bodů displeje, pro který je daná sada drawable primárně určena. Například pro telefony se střední hustotou bodů displeje je určen adresář „drawable-mdpi“.

Tento návrh umožňuje vývojářům vložit do aplikace .png obrázky a další části grafiky v několika různých rozlišeních pro různé hustoty bodů displeje. Toto ovšem není nutností. V případě, že pro pro dané zařízení není k dispozici drawable v příslušném adresáři, je vybráno drawable z jiného vhodného adresáře a následně je přeškálováno na požadované rozlišení. Pouze je tato operace lehce výpočetně náročnější, a při jejím využití pro všechna rozlišení displeje nemusí mít aplikace vždy pixel-perfect grafiku uživatelského rozhraní.

Podadresář layout

Podadresář layout obsahuje xml soubory, v kterých je popsán layout(rozložení) uživatelského rozhraní jednotlivých Activity a dalších grafických prvků aplikace.

V případě potřeby je možné vytvořit další podadresáře, do kterých je možné umístit upravený layout pro různé skupiny mobilních zařízení, nebo pro odlišnou orientaci obrazovky. Pro obrazovku v landscape režimu je možné vytvořit podadresář „layout-land“. Případně pro specifické rozměry displeje je také možné vytvořit specializovaný podadresář. Například pro tablety s úhlopříčkou

7 palců a větší (šířka displeje 600 a více dp) je možné vytvořit podadresář „layout-w600dp“.

Podadresář values

Podadresář values obsahuje xml soubory, které slouží k ukládání hodnot a informací, které jsou v aplikaci použity jako konstanty. Tyto xml soubory slouží, jako centrální úložiště daného typu hodnot nebo informací pro celou aplikaci. Neboť není dobrou vývojářskou praxí tyto informace ukládat natvrdo v Java kódu aplikace z důvodu možnosti centralizované správy uložených hodnot.

Soubor strings.xml je soubor, který slouží k uložení textů, které jsou zobrazeny v uživatelském rozhraní aplikace.

Soubor colors.xml je soubor, který slouží k deklaraci barev, které jsou zobrazeny v uživatelském rozhraní aplikace.

Soubor dimens.xml je soubor, který slouží k deklaraci rozměrů, které jsou použity v layoutu(rozložení) uživatelského rozhraní aplikace.

Soubor styles.xml je soubor, který slouží k deklaraci vlastních stylů, které je možné dále použít pro prvky uživatelského rozhraní aplikace.

Je možné vytvořit několik podadresářů s názvem začínajícím prefixem „values“ a končícím kódem jazyka, pro který jsou určeny. Například podadresář „values-fr“ pro hodnoty, které se použijí, pokud je jazyk aplikace nastaven na francouzštinu. Díky tomuto konceptu je možné snadno do aplikace přidávat další jazykové mutace.

3.6 Možnosti šíření aplikace

Po provedení buildu výsledné aplikace je vytvořen .apk archiv, který obsahuje .dex soubor s bytecode aplikace, data původně obsažená v adresáři res vývojového prostředí Eclipse s dalšími podpůrnými soubory a soubor AndroidManifest.xml, kde jsou uvedeny informace potřebné pro ověření kompatibility aplikace s daným zařízením a seznam oprávnění, která je třeba aplikaci při instalaci povolit.

Po stažení aplikace do mobilního zařízení a následném spuštění instalace je třeba povolit případný seznam oprávnění, která aplikace vyžaduje pro svůj běh. Od Android verze 6.0 (verze vývojářského API 23) je možné dodatečně zakázat některá oprávnění, která byla aplikaci při instalaci udělena (23).

Aplikaci je do telefonu možné nainstalovat několika způsoby.

3.6.1 Instalace z obchodu Google Play

Internetový obchod Google Play (dříve Android Market) je hlavním a oficiálním distribučním kanálem pro šíření aplikací pro OS Android. Jak již název napovídá tento obchod patří společnosti Google Inc. Do tohoto obchodu mohou přistupovat zařízení s OS Android ve verzi 2.2 a vyšším. A od této verze

mají téměř všechna zařízení s OS Android předinstalovanou také aplikaci internetového obchodu Google Play. S použitím tohoto obchodu je tedy možné dostat aplikaci téměř ke všem uživatelům mobilních zařízení s OS Android.

Pro možnost distribuce aplikací přes internetový obchod Google Play je třeba si založit vývojářský účet. Založení tohoto účtu stojí jednorázový poplatek 25USD (22).

3.6.2 Individuální distribuce archivu s aplikací

Narozdíl od operačních systémů iOS společnosti Apple a Windows Phone společnosti Microsoft umožňuje operační systém Android instalaci aplikací třetích stran z neznámých zdrojů bez nutnosti instalace těchto aplikací z oficiálního obchodu.

Pro umožnění instalace aplikací třetích stran z neznámých zdrojů je nutné v nastavení telefonu povolit možnost „Povolit instalaci aplikací z neznámých zdrojů“. A následně je možné instalovat aplikace pomocí apk archivu staženého například z internetových stránek tvůrce aplikace.

Tento způsob šíření aplikace je ovšem vhodný pouze pro aplikace, které jsou určeny pro uzavřený okruh uživatelů, jako jsou například vnitropodnikové aplikace.

3.6.3 Internetové obchody třetích stran

Další možností šíření aplikace jsou internetové obchody společností třetích stran. Tyto obchody nemají takový tržní podíl, jako oficiální obchod Google Play. Ovšem nabízejí zajímavou alternativu, neboť neobsahují tak velké množství aplikací, a tudíž i konkurence v těchto obchodech je na mnohem nižší úrovni, než na oficiálním Google Play.

Navíc většina internetových obchodů třetích stran nevyžaduje žádný registrační poplatek k vytvoření vývojářského účtu, jako oficiální obchod Google Play. Mezi nejvýznamnější z těchto obchodů patří Amazon Appstore for Android, Samsung Galaxy Apps nebo Opera Mobile Store.

3.7 Možnosti zpeněžení aplikace

Cílem většiny vývojářů mobilních aplikací je dosažení zisku z vytvářené mobilní aplikace. Existuje několik běžně používaných finančních modelů, které jsou nejčastěji používány. K maximalizaci zisku bývá často použita i kombinace několika modelů.

Již při prvotní analýze mobilní aplikace je třeba důkladně zvážit model zpeněžení aplikace a všechny klady a zápory, které přináší. Z nich vyplyne míra rizika a odhad návratnosti investice, která bude do vývoje aplikace vložena.

3.7.1 Zakoupení aplikace v obchodu Google Play

Platba za stažení aplikace z obchodu Google Play je nejjednodušším modelem zpoplatnění. Není třeba provádět žádné další úpravy v samotné aplikaci. Platba je provedena na úrovni obchodu Google Play, který si za toto účtuje poplatek ve výši 30% inzerované ceny.

Tento model zpeněžení je vhodný pro typ aplikace, kde neexistuje žádná významná konkurence. Uživatel, který tento typ aplikace potřebuje, si tuto aplikaci s vysokou pravděpodobností opravdu zakoupí.

Nevýhodou tohoto modelu je skutečnost, že před samotným zakoupením aplikace uživatel vidí pouze popis aplikace v obchodu Google Play a přiložené screenshoty, případně video. Na základě těchto informací se uživatel rozhoduje, zda aplikaci zakoupit, a nemá možnost otestovat si samotnou aplikaci.

3.7.2 Model základní a placené verze

Tento model zpeněžení je v obchodu Google Play vývojáři často používán. Spočívá ve vytvoření dvou verzí aplikace.

Základní verze aplikace je uživatelům k dispozici zdarma ke stažení, avšak její možnosti použití jsou částečně omezeny. Toto omezení může spočívat v zablokování některých funkcí aplikace, v omezeném počtu použití některých funkcí, v nucené časové prodlevě při startu aplikace, nebo v přítomnosti reklam. Tato základní verze umožňuje uživateli vyzkoušet si funkčnost aplikace, a pokud mu vyhovuje, zakoupit si placenou verzi, která je bez popsaných omezení.

3.7.3 Platby v rámci aplikace (In-App Purchases)

Platby v rámci aplikace fungují na principu, kdy samotné stažení aplikace z obchodu Google Play je typicky zdarma, ale přímo během používání aplikace je možné zakoupit například odemčení Pro verze aplikace, nebo dokupovat jednotlivé součásti a rozšíření aplikace.

Tento model umožňuje uživateli také vyzkoušení funkčnosti aplikace, a následné rozhodnutí, zda si zakoupit Pro verzi aplikace, případně dokoupit některá rozšíření.

Platby v rámci aplikace jsou také hojně využívány vývojáři her pro mobilní telefony, kde pomocí plateb v rámci aplikace umožňují nákup různých předmětů ve hře, případně nákup virtuální herní měny.

3.7.4 Reklamy v mobilních aplikacích

Použití reklam v mobilních aplikacích je velmi často používaným modelem zpeněžení. Uživatel si aplikaci z obchodu Google Play může stáhnout zdarma. Ovšem během jejího používání jsou mu zobrazeny reklamy.

Existují dva možné modely zisku z reklamy. Prvním modelem je zisk za kliknutí uživatele na danou reklamu. V tomto případě vývojář aplikace následně získává procentuální část ceny, kterou inzerent zaplatil za reklamu.

Druhým modelem je zisk ze zobrazení reklamy. V případě tohoto modelu vývojář získává určitou paušální částku za daný počet zobrazení reklamy bez ohledu na počet kliknutí na reklamu uživatelem.

Použití reklam v aplikaci je často kombinováno s modelem základní a placené verze, kdy v základní verzi jsou uživateli zobrazeny reklamy. Pro verze následně umožňuje uživateli odstranění reklam. Případně jsou reklamy v aplikaci také často kombinovány s platbami v rámci aplikace, kdy je odstranění reklam umožněno uživateli pomocí jednorázové platby v rámci aplikace.

Společnost Google pro zobrazení reklam v mobilních aplikacích poskytuje svou službu Google AdMob, která je vývojáři mobilních aplikací často využívána. Existují ovšem i další poskytovatelé reklam pro mobilní aplikace. Například MobClix, MobFox, Airpush a další(13).

Kapitola 4

Praktická část

Praktická část obsahuje prvotní analýzu aplikace, instalaci vývojového prostředí, vývoj a architekturu aplikace, návrh uživatelského rozhraní aplikace a následné umístění do obchodu Google Play. Spolu s výsledným zhodnocením vytvořené aplikace po několika měsících nasazení v reálném provozu.

4.1 Vývoj aplikace Sensors Toolbox

Na základě analyzovaných teoretických poznatků si praktická část klade za cíl vytvořit a publikovat do obchodu Google Play aplikaci Sensors Toolbox pro OS Android. Tato aplikace umožní uživateli zobrazovat běžně nepřístupná data z vybraných senzorů, informace o stavu Wi-Fi připojení, mobilní sítě a baterie. Uživatelské rozhraní aplikace bude vytvořeno s ohledem na principy Material design. Aplikace je koncipována, jako offline aplikace, která pro svou funkci nevyžaduje nutně internetové připojení.

4.1.1 Výběr vývojového prostředí

V části 3.3.2 jsou podrobně popsána oficiální vývojová prostředí pro tvorbu aplikací pro OS Android spolu s jejich výhodami a nevýhodami. Vývoj mobilní aplikace probíhal na PC, které bylo vybaveno slabším hardware (procesor Intel Celeron E1200 a operační paměť o velikosti 2GB). Kvůli vysokým hardwarovým nárokům vývojového prostředí Android Studio bylo pro vývoj mobilní aplikace zvoleno vývojové prostředí Eclipse.

4.1.2 Instalace vývojového prostředí Eclipse s ADT pluginem

Instalace vývojového prostředí Eclipse a následná integrace ADT pluginu se stává z několika kroků.

Prvním krokem je stažení samotného vývojového prostředí ze stránek Eclipse Foundation „eclipse.org/downloads/“. Vývojové prostředí je distribuováno ve formě

.zip archivu, který je třeba následně extrahovat. Po extrahování je vývojové prostředí připraveno ke spuštění.

Vývojové prostředí Eclipse ke svému běhu vyžaduje nainstalované JDK (Java Development Kit). V případě, že na PC není JDK zatím nainstalováno, je možné jej stáhnout a následně nainstalovat ze stránek společnosti Oracle na adrese „http://www.oracle.com/technetwork/java/javase/downloads/“.

Při prvním spuštění vývojového prostředí je třeba specifikovat adresář, ve kterém bude umístěno tzv. workspace. Do tohoto adresáře budou ukládány projekty, které budou ve vývojovém prostředí Eclipse vytvořeny v rámci daného workspace.

Po vytvoření workspace a spuštění vývojového prostředí je třeba nainstalovat ADT plugin. Tato instalace se provede pomocí stisknutí tlačítka „Help“ na horní liště, a následné volby „Install New Software...“. Po zobrazení dialogu je třeba do pole pro url adresu zadat „https://dl-ssl.google.com/android/eclipse/“ a následně zvolit instalaci celého balíčku „Developer Tools“ a provést instalací pomocí tlačítka „Next“ a „Install“.

4.1.3 Analýza problematiky a odvětví

Mobilní telefony s OS Android jsou vybaveny celou řadou různých senzorů a čidel, která jsou k dispozici vývojářům mobilních aplikací. Ovšem samotný OS Android neumožňuje k mnoha údajům z těchto čidel a senzorů přímý přístup uživateli bez použití aplikací třetích stran.

V minulosti bylo vytvořeno již několik aplikací, které se snažily shrnout pro uživatele různými způsoby data z těchto senzorů. Tyto aplikace však většinou vznikly v době, kdy nejnovější verzí OS Android byla verze 2.3. A do dnešních dnů se nedočkaly větších aktualizací. Tomuto odpovídá návrh uživatelského rozhraní aplikací a zároveň také limity jejich schopností.

Zároveň žádná z těchto aplikací plně nevyužívá potenciál integrace dalších souvisejících nástrojů, jako je například možnost naskenování seznamu zařízení připojených ve Wi-Fi síti, která by byla dostupná spolu se stávajícími informacemi o stavu Wi-Fi sítě.

4.1.4 Analýza návrhu aplikace

Ze zadání byl vytvořen seznam funkčních a nefunkčních požadavků na aplikaci. Tyto požadavky je třeba analyzovat a na jejich základě navrhnout řešení aplikace.

Mezi funkční požadavky patří možnost zobrazení podrobných informací o stavu baterie, mobilního připojení, Wi-Fi sítě, GPS, akcelerometru, magnetometru a senzoru vzdálenosti. Dalším funkčním požadavkem je možnost zobrazení kompasu na displeji telefonu, pokud mobilní telefon bude obsahovat potřebné senzory.

Aplikace bude navržena tak, aby pro každou z těchto kategorií funkčních požadavků obsahovala samostatnou obrazovku.

Mezi uživateli skryté funkční požadavky patří intergrace analytických nástrojů pro sledování způsobu, jakým uživatelé využívají aplikaci, a nástrojů pro monitorování spolehlivosti aplikace a jejích případných pádů.

Pro sledování způsobu využití aplikace bude do aplikace integrován nástroj Google Analytics. Pro sledování spolehlivosti aplikace a jejich případných pádů bude do aplikace implementována knihovna Splunk MINT.

Mezi nefunkční požadavky patří nejnižší podporovaná verze OS Android 4.0. Z tohoto požadavku vyplývá nutnost použití Android Support v4 knihovny pro některé prvky uživatelského rozhraní, které nejsou přímo v SDK pro Android 4.0 integrovány.

Dalším nefunkčním požadavkem je plná funkčnost aplikace z hlediska uživatele i v offline módu. Tento požadavek je zajištěn návrhem aplikace, pouze analytická data do knihoven Google Analytics s Splunk MINT nebudou odesílána průběžně, ale až po opětovném připojení aplikace k Internetu.

4.1.5 Návrh uživatelského rozhraní aplikace

Uživatelské rozhraní aplikace je navrženo s využitím několika Activity, mezi kterými může uživatel přecházet. Navigace mezi jednotlivými Activity je zajištěna pomocí prvku uživatelského rozhraní tzv. Action Bar. Tedy pomocí navigační lišty, která je umístěna v horní části obrazovky a zobrazuje název aktuálně zobrazené obrazovky a případně tlačítko pro návrat zpět na předchozí Activity.

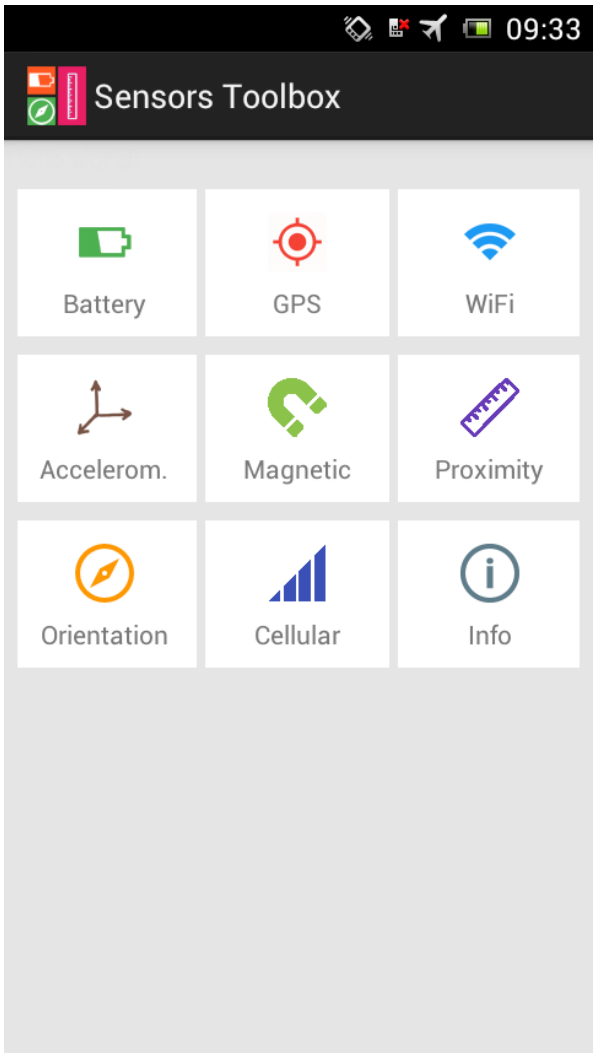
Celé grafické provedení uživatelského rozhraní aplikace je koncipováno s ohledem na zásady návrhu s využitím Material design.

Uživatelské rozhraní aplikace je navrženo tak, aby na všechny hlavní obrazovky (Activity) bylo možné se dostat pomocí jediného stisknutí tlačítka. Z tohoto důvodu domácí obrazovka aplikace (MainActivity) obsahuje dlaždice srovnané v mřížce pomocí prvku uživatelského rozhraní Grid View. Každá dlaždice směřuje na jednu z hlavních Activity aplikace. Vzhled domácí obrazovky aplikace je zachycen na obrázku 4.1.

Po stisknutí libovolné z dlaždic dojde k zobrazení příslušné Activity s podrobnými informacemi o daném senzoru, části mobilního telefonu, nebo nástroji. Pouze poslední dlaždice s názvem „Info“ zobrazí obrazovku s informacemi o aplikaci.

4.1.6 Podrobné informace o stavu baterie

Obrazovka s podrobnými informacemi o stavu baterie (BatteryActivity) obsahuje mimo jiné i informace o teplotě a napětí baterie, ke kterým se z uživatelského rozhraní OS Android není přímo možné dostat. Z této obrazovky je také pomocí tlačítka „Open usage summary“ možné otevřít obrazovku nas-



Obrázek 4.1: Domácí obrazovka aplikace Sensors Toolbox s dlaždicemi

tavení OS Android s informacemi o využití baterie jednotlivými aplikacemi. Tato obrazovka nastavení je otevřena pomocí volání následujícího Intent:

```
Intent i = new Intent(Intent.ACTION_POWER_USAGE_SUMMARY);
startActivity(i);
```

Samotné zaregistrování Activity pro příjem zpráv o změně stavu baterie je provedeno následujícím způsobem:

```
IntentFilter if = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
Intent intent = registerReceiver(null, if);

BatteryInfoItem i = new BatteryInfoItem();

// status
int status = intent.getIntExtra(BatteryManager.EXTRA_STATUS, Const.INT_EMPTY);
if(status == BatteryManager.BATTERY_STATUS_UNKNOWN){
    i.status = getResources().getString(R.string.battery_status_unknown);
} else if (status == BatteryManager.BATTERY_STATUS_CHARGING) {
    i.status = getResources().getString(R.string.battery_status_charging);
} else if (status == BatteryManager.BATTERY_STATUS_DISCHARGING) {
        i.status = getResources().getString(
        R.string.battery_status_discharging);
} else if (status == BatteryManager.BATTERY_STATUS_NOT_CHARGING) {
        i.status = getResources().getString(
        R.string.battery_status_not_charging);
} else if (status == BatteryManager.BATTERY_STATUS_FULL) {
        i.status = getResources().getString(R.string.battery_status_full);
}

...
```

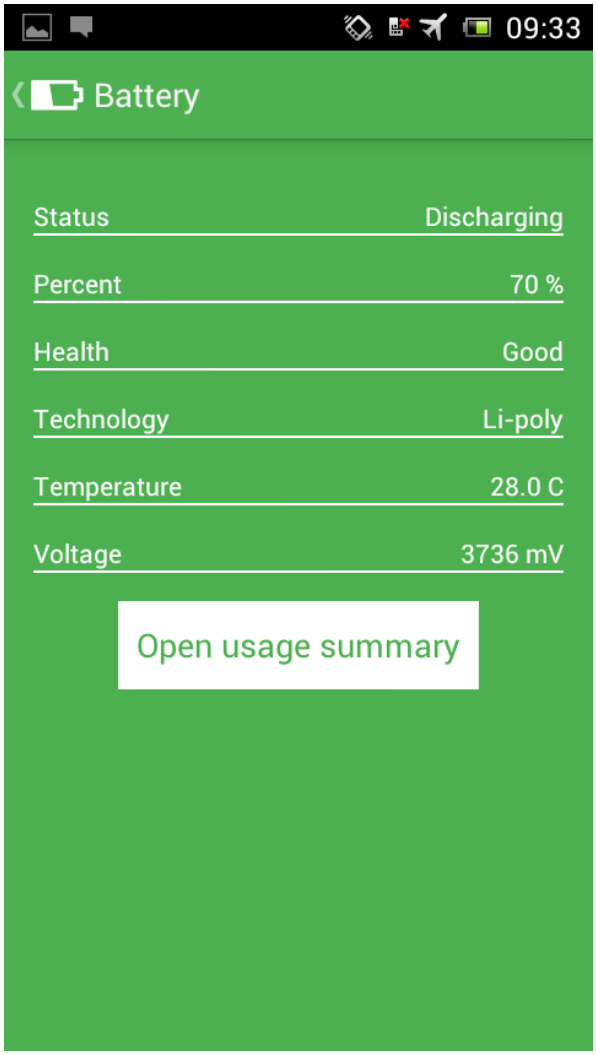
Přičemž „BatteryInfoItem“ je vlastní kontejner, do kterého jsou ukládány informace zjištěné z „Intent“. Neboť Intent obsahuje pouze číslený kód stavu baterie, je třeba mu přiřadit příslušný text čitelný pro uživatele. Texty jsou uloženy v souboru „strings.xml“ a jejich přiřazení do proměnné „status“ typu „String“ probíhá pomocí volání metody „getResources().getString(...)“ ve výše uvedené ukázce.

4.1.7 Podrobné informace o GPS

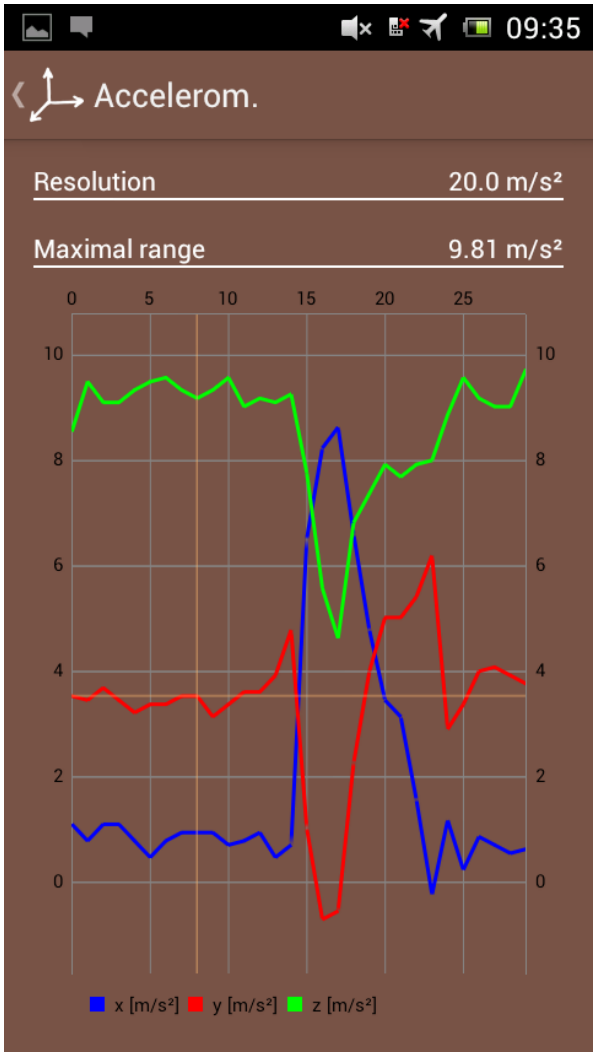
Obrazovka s podrobnými informacemi o GPS (GpsActivity) obsahuje mimo jiné informace o nadmořské výšce (pokud byla GPS poloha získána za pomoci čtyřech a více satelitů), o celkovém počtu satelitů v dohledu a o celkovém počtu satelitů, které byly použity pro získání GPS polohy. Zároveň obrazovka obsahuje sloupcový graf, kde jsou obsaženy jednotlivé satelity v dohledu a znázorněna síla jejich signálu.

4.1.8 Podrobné informace o Wi-Fi

Obrazovka s podrobnými informacemi o Wi-Fi obsahuje mimo jiné BSSID (Basic service set identifier) přístupového bodu, ke kterému je mobilní zařízení připojeno (typicky se jedná o MAC adresu). Dále obsahuje hodnotu RSSI (Received signal strength indicator), což je indikátor síly přijímaného Wi-Fi signálu. Tato obrazovka také obsahuje IP adresu a MAC adresu mobilního zařízení.



Obrázek 4.2: Podrobné informace o stavu baterie



Obrázek 4.3: Podrobné informace o akcelerometru

4.1.9 Podrobné informace o akcelerometru

Obrazovka s podrobnými informacemi o akcelerometru obsahuje kromě aktuálních naměřených hodnot také graf s průběhem jednotlivých hodnot v čase. Dále je zde možné zjistit označení a jméno výrobce akcelerometru spolu s jeho rozlišením a rozsahem v m/s^2.

4.1.10 Podrobné informace o magnetometru

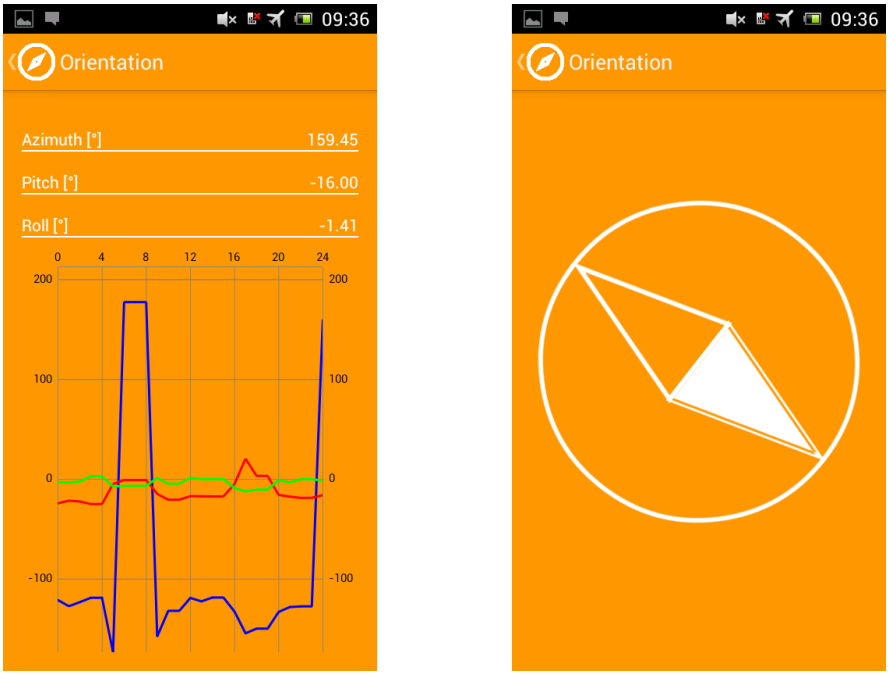
Obrazovka s podrobnými informacemi o magnetometru obsahuje aktuálně naměřené hodnoty sil působení magnetického pole ve směru jednotlivých os spolu s výslednou silou působení magnetického pole v μT. Dále je zde také

Obrázek 4.4: Podrobné informace o magnetometru

možné zjistit označení a jméno výrobce magnetometru spolu s jeho rozlišením a maximálním rozsahem. Obdobně, jako na obrazovce s podrobnými informacemi o akcelerometru, je zde také zobrazen graf s průběhem jednotlivých hodnot v čase.

4.1.11 Podrobné informace o senzoru přiblížení

Mobilní telefony s OS Android obsahují zabudovaný senzor přiblížení z důvodu, že v situaci, kdy má uživatel telefon u ucha při telefonování, je třeba deaktivovat dotykovou vrstvu displeje, případně displej vypnout dočasně celý. A to z důvodu předcházení nechtěným stiskům ovládacích prvků na obra-



Obrázek 4.5: Podrobné informace o prostorové orientaci

zovce tváří během telefonování. Tento senzor typicky vrací pouze informaci o tom, zda je v blízkosti displeje nějaký předmět, nebo nikoliv.

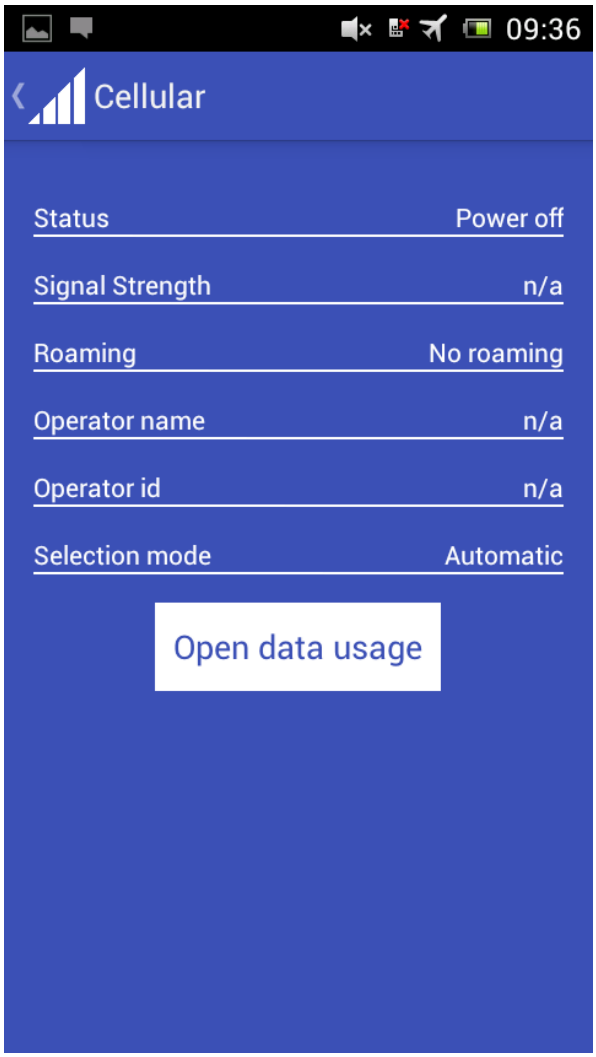
Obrazovka s podrobnými informacemi o senzoru přiblížení zobrazuje aktuální naměřenou hodnotu, jméno senzoru a jeho výrobce. Dále je možné zobrazit rozlišení a rozsah senzoru v cm spolu s proudem potřebným pro provoz senzoru při jeho používání v mA.

4.1.12 Obrazovka s informacemi o prostorové orientaci

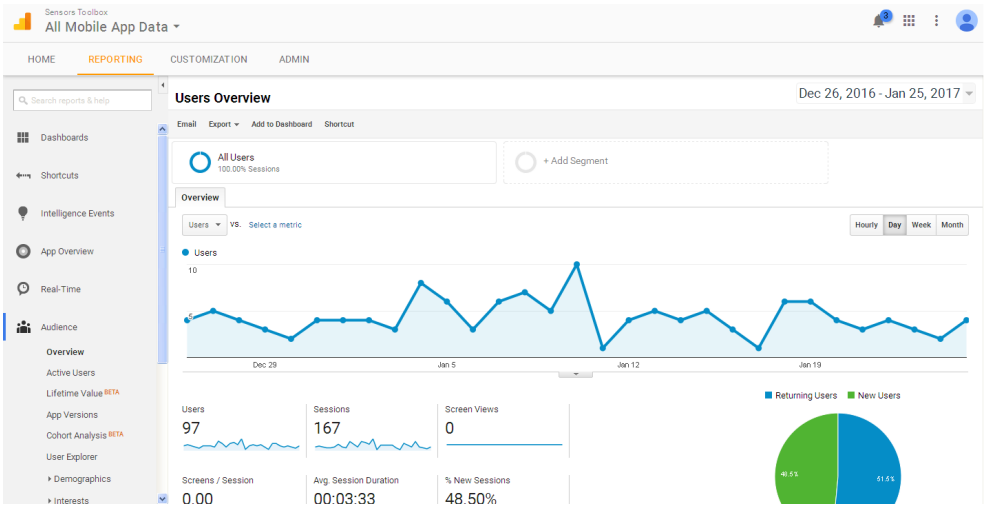
Tato obrazovka využívá dat z akcelerometru a magnetometru pro zjištění azimutu (Azimuth), úhlu klopení (Pitch) a úhlu klonění (Roll). Průběh jednotlivých hodnot v čase je zaznamenáván do grafu. Na základě zjištěného azimutu aplikace zobrazuje také vektorově vykreslený kompas.

4.1.13 Obrazovka s informacemi o mobilním připojení

Tato obrazovka zobrazuje informace o síti mobilního operátora, ke které je mobilní telefon připojen. Jsou zde k dispozici podrobné informace, jako síla signálu v dBm, informace o roamingu, název a id operátora a informace o způsobu volby operátora při roamingu. Dále je zde k dispozici tlačítko, které zobrazí obrazovku s informacemi o využití dat v nastavení telefonu.



Obrázek 4.6: Podrobné informace o mobilním připojení



Obrázek 4.7: Webové rozhraní Google Analytics(Zdroj:Autor)

4.1.14 Sledování způsobu využití aplikace

Pro sledování způsobu využití aplikace uživateli byla do aplikace integrována knihovna Google Analytics.

4.1.15 Sledování spolehlivosti aplikace a jejich případných pádů

Pro sledování spolehlivosti aplikace a jejich případných pádů byla do aplikace integrována knihovna Splunk MINT.

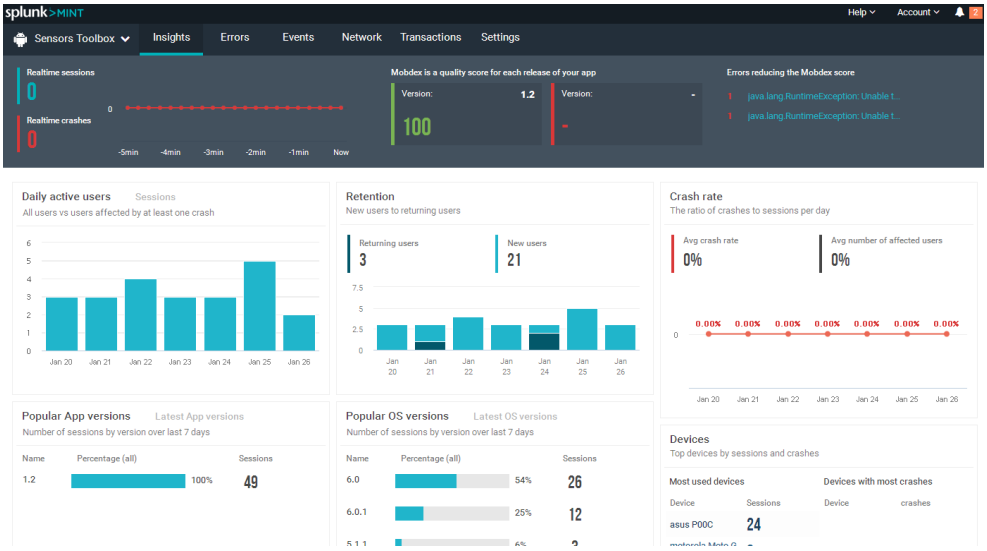
4.1.16 Systém pro správu verzí

Systémy pro správu verzí jsou primárně určeny pro podporu vývoje softwarových aplikací v týmu více lidí, kde umožňují verzovat jednotlivé soubory a spravovat jejich změny tak, aby se dostaly ke všem členům týmu.

Ovšem i při vývoji této aplikace bylo užitečné využít systém pro správu verzí pro přehledné verzování jednotlivých úprav zdrojového kódu a zálohování zdrojových kódů aplikace na server.

S současné době existují dva majoritní verzovací systémy SVN (Apache Subversion) a GIT (Global Information Tracker).

Při vývoji této mobilní aplikace byl použit verzovací systém GIT. Primárním důvodem pro toto rozhodnutí byla možnost případného provozu systému GIT i lokálně bez vzdáleného serveru (toto systém SVN neumožňuje). I když je systém GIT ve většině případů využíván spolu se serverem, je mnohem flexibilnější při práci v momentě, kdy není dostupné připojení k síti Internet.



Obrázek 4.8: Webové rozhraní nástroje Splunk MINT(Zdroj:Autor)

Princip verzovacího systému GIT je takový, že poté co vývojář provede úpravu kódu, provede následně tzv. „commit“, do kterého přidá změny, které provedl. A následně pomocí příkazu „push“ tyto změny odešle na server. V případě, že si vývojář chce ze serveru stáhnout změny, od ostatních členů týmu provede toto pomocí příkazu „pull“.

Nejdůležitější příkazy verzovacího systému GIT jsou uvedeny v následující ukázce:

```
// naklonování obsahu GIT repozitáře ze serveru do lokálního PC
git clone <adresa_vzdaleneho_serveru>

// přidání souboru do commitu
git add <jmeno_souboru>

// provedení commitu
git commit -a -m "<slovni_popis_commitu>"

// odeslání všech lokálně uložených commitů na vzdálený server
git push

// stažení změn ze vzdáleného serveru
git pull
```

4.1.17 Testování aplikace

Testování aplikace před vystavením v obchodu Google Play probíhalo za pomoci několika testovacích uživatelů. Nalezené nedostatky uživatelského rozhraní a drobné chyby byly opravovány v průběhu testování.

Samotné testování probíhalo na následujících mobilních zařízeních:

- Sony Ericsson Xperia ray (ST18i) (verze OS Android 4.0.4)

- Samsung Galaxy S III Mini (GT-I8190) (verze OS Android 4.2)
- Samsung Galaxy S III Mini VE (GT-I8200) (verze OS Android 4.2)
- ALCATEL ONETOUCH 5022D POP STAR Dual SIM (verze OS Android 5.0)
- Sony Xperia Z3 compact D5803 (verze OS Android 6.0.1)
- Samsung Galaxy Tab 3 Lite 7.0 (verze OS Android 4.2)

4.1.18 Vystavení aplikace v obchodě Google Play

Pro možnost publikování aplikace v obchodě Google Play je třeba se zaregistrovat na adrese `https://play.google.com/apps/publish/signup/` a zaplatit jednorázový registrační poplatek 25 USD.

Po úspěšné registraci je třeba založit název aplikace v rozhraní obchodu. Následně je třeba nastavit hodnocení obsahu aplikace, vyplnit popis aplikace, nahrát screenshoty z aplikace pro různá rozlišení a následně nahrát samotný podepsaný .apk archiv aplikace. Při tomto nahrávání obchod Google Play kontroluje technické podmínky, které je třeba splnit. V případě, že nahrávaný .apk archiv nesplňuje některé podmínky, obchod Google Play zobrazí podrobné informace i s popisem, jak tento problém odstranit.

Po splnění všech podmínek je možné publikovat aplikaci do Google Play. Tato publikace je automatická (není vyžadována žádná forma manuální kontroly člověkem, jako je tomu v případě AppStore společnosti Apple Inc.). V řádu několika hodin po odeslání k publikaci je tedy aplikace dostupná uživatelům ke stažení v samotném obchodě Google Play.

Tato aplikace je dostupná ke stažení na internetovém obchodě Google Play na adrese `https://play.google.com/store/apps/details?id=org.beckamichal.sensorstoolbox`.

4.2 Zhodnocení výsledků a možnosti dalšího vývoje

V praktické části této diplomové práce se úspěšně podařilo provést prvotní analýzu aplikace Sensors Toolbox, instalaci vývojového prostředí, návrh architektury aplikace a jejího uživatelského rozhraní spolu s vývojem samotné aplikace a její následné umístění do obchodu Google Play.

Aplikace je schopna získávat údaje z nejvíce rozšířených senzorů v mobilních telefonech s OS Android a v přehledné formě je zobrazovat uživateli.

První verze aplikace byla do obchodu Google Play publikována 22.9.2016 od tohoto data do začátku února 2017 byly vydány celkem tři verze aplikace, které postupně na základě údajů z nástroje Splunk MINT a na základě průběžné zpětné vazby uživatelů přinášely drobné opravy a vylepšení.

Na základě údajů z analytického nástroje Google Analytics na počátku února 2017 aplikaci aktivně používalo více, než 100 uživatelů při celkovém počtu více, než 200 instalací. Vzhledem k tomu, že tato aplikace nebyla nijak propagována, a vzhledem ke specifické skupině uživatelů, na kterou je aplikace zamířena, lze tento výsledek považovat za úspěšný. Rovněž v obchodě Google Play je aplikace hodnocena uživateli pozitivně.

Další vývoj aplikace by bylo vhodné směřovat k rozšíření uživatelské základny pomocí přidávání podpory méně častých senzorů do aplikace a pomocí propagace aplikace na odborných fórech a diskusích. Po získání širší uživatelské základny se nabízí možnost zpeněžení aplikace pomocí modelu základní verze zdarma, případně obsahující reklamy v aplikaci, a rozšířené placené verze, která by umožňovala odstranění případných reklam a některé pokročilejší funkce oproti základní verzi.

Závěr

V úvodu teoretické části této diplomové práce byl stručně popsán vývoj mobilních telefonů směřující přes telefony schopné spouštět aplikace třetích stran (J2ME) až po telefony s operačním systémem. Nejrozšířenější konkurenční mobilní operační systémy jsou zde zároveň také charakterizovány spolu se svými výhodami a nedostatky. Teoretická část této práce je zaměřena převážně na charakteristiku historie, specifik vývoje a možností telefonů s operačním systémem Android.

Druhá polovina teoretické části je věnována možnostem vývoje aplikací pro zařízení s OS Android, analýze vývojového prostředí a základní struktury aplikace. Závěr teoretické části uvádí možnosti zpeněžení a šíření aplikace.

Poznatky získané v teoretické části byly spolu s vlastními zkušenostmi z oblasti vývoje mobilních aplikací použity v praktické části této práce. V praktické části byla navržena a následně vytvořena mobilní aplikace Sensors Toolbox umožňující uživatelsky přehledné zobrazení údajů o Wi-Fi a mobilním připojení, parametrech baterie a dat z nejběžnějších senzorů v telefonech s OS Android.

Aplikace byla testována s pomocí několika testovacích uživatelů na různých zařízeních. Následně byla aplikace vystavena do obchodu Google Play. Na základě zpětné vazby a údajů z nástrojů pro sledování spolehlivosti aplikace byly vydány do obchodu Google Play postupně tři verze aplikace s drobnými opravami a vylepšeními chování aplikace.

V závěru praktické části je zhodnocen celkový úspěch aplikace a možnosti jejích dalšího vývoje a zpeněžení.

Tato diplomová práce tedy charakterizuje možnosti dynamicky se rozvíjejícího trhu s mobilními telefony s OS Android. Spolu s analýzou možností vývoje aplikací pro tuto platformu a následným vývojem aplikace a jejím publikováním do obchodu Google Play.

Seznam použitých zdrojů

(1) BEČKA, Michal. *Přenesení hry Heroes of Might and Magic III na platformy J2ME a Android.* Praha, 2013. Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií, Katedra softwarového inženýrství. Vedoucí Bakalářské práce Ing. Miroslav Balík, Ph.D.

(2) ANDROID HEADLINES. *The Android Revolution* [online] [cit. 2016-11-22]. Dostupné z: `http://www.androidheadlines.com/2013/10/android-revolution-htc-dream-became-1-billion-activations-years-later.html`

(3) APPLE. *iTunes Xcode* [online] [cit. 2016-11-20]. Dostupné z: `https://itunes.apple.com/cz/app/xcode/id497799835?mt=12`

(4) ARS TECHNICA. *Microsoft brings Android, iOS apps to Windows 10* [online] [cit. 2016-11-17]. Dostupné z: `http://arstechnica.com/information-technology/2015/04/microsoft-brings-android-ios-apps-to-windows-10/`

(5) BAAR GROUP. *KVM: A Small Java Virtual Machine for J2ME*[online] [cit. 2016-9-28]. Dostupné z: `http://www.barrgroup.com/Embedded-Systems/How-To/KVM-J2ME-Java-Virtual-Machine`

(6) BLACKBERRY LIMITED. *Amazon AppStore now available for your BlackBerry 10 smartphone* [online] [cit. 2016-10-5]. Dostupné z: `http://us.blackberry.com/apps/amazon-appstore.html`

(7) BLACKBERRY LIMITED. *Research In Motion Changes Its Name to BlackBerry*[online] [cit. 2016-10-5]. Dostupné z: `http://press.blackberry.com/en/press/2013/research-in-motion-changes-its-name-to-blackberry.html`

(8) BUCHANAN, Ricky. *iPhones 1, 4, 5, 6 and 6 Plus chart* [online] [cit. 2016-11-20]. Dostupné z: http://atmac.org/contents/uploads/iphones-1-to-6-900x358.jpg

(9) CRACKBERRY. *BlackBerry 9000 Smartphone Review* [online] [cit. 2016-10-5]. Dostupné z: http://crackberry.com/blackberry-9000-smartphone-review-new-eye-candy

(10) DE HERRERA, Chris. *Windows CE 2.0 Color Handheld PC Screen Shots* [online] [cit. 2016-10-17]. Dostupné z: http://www.pocketpcfaq.com/wce/wce20scrn.htm

(11) EPSTEIN, Zach. *Apple and Google dominate smartphone space* [online] [cit. 2016-10-17]. Dostupné z: http://bgr.com/2011/12/13/apple-and-google-dominate-smartphone-space-while-other-vendors-scramble/

(12) FIDR. *iOS 9: The information for iPhone and iPad* [online] [cit. 2016-11-20]. Dostupné z: http://www.freeiphonedatarecovery.com/news/ios-9-iphone-ipad-rcm992x860.jpg

(13) GOOGLE INC. *AdMob by Google* [online] [cit. 2016-11-29]. Dostupné z: https://www.google.com/admob/

(14) GOOGLE INC. *ADT Plugin Release Notes* [online] [cit. 2016-11-27]. Dostupné z: https://developer.android.com/studio/tools/sdk/eclipse-adt.html

(15) GOOGLE INC. *Android Dashboards* [online] [cit. 2016-11-22]. Dostupné z: https://developer.android.com/about/dashboards/index.html

(16) GOOGLE INC. *Android Developer Console* [online] [cit. 2016-11-27]. Dostupné z: https://developer.android.com/training/basics/activity-lifecycle/starting.html

(17) GOOGLE INC. *Android Developer Console: Fragment* [online] [cit. 2016-11-29]. Dostupné z: https://developer.android.com/guide/components/fragments.html

(18) GOOGLE INC. *Android history* [online] [cit. 2016-11-22]. Dostupné z: https://www.android.com/history/

(19) GOOGLE INC. *Android NDK Developers* [online] [cit. 2016-11-27]. Dostupné z: https://developer.android.com/ndk/index.html

(20) GOOGLE INC. *Android Studio* [online] [cit. 2016-11-27]. Dostupné z: https://developer.android.com/studio/index.html

(21) GOOGLE INC. *Building a Flexible UI* [online] [cit. 2016-11-29]. Dostupné z: https://developer.android.com/training/basics/fragments/fragment-ui.html

(22) GOOGLE INC. *How to use the Google Play Developer Console* [online] [cit. 2016-11-29]. Dostupné z: https://support.google.com/googleplay/android-developer/answer/6112435?hl=en

(23) GOOGLE INC. *System Permissions* [online] [cit. 2016-11-29]. Dostupné z: https://developer.android.com/guide/topics/security/permissions.html

(24) HOWTOGEEK. *Android is Based on Linux, But What Does That Mean?* [online] [cit. 2016-11-22]. Dostupné z: http://www.howtogeek.com/189036/android-is-based-on-linux-but-what-does-that-mean/

(25) HPC FACTOR. *The History of Windows CE* [online] [cit. 2016-10-17]. Dostupné z: http://www.hpcfactor.com/support/windowsce/

(26) JAVA T POINT. *Android Activity lifecycle* [online] [cit. 2016-11-27]. Dostupné z: http://www.javatpoint.com/android-life-cycle-of-activity

(27) MAHANTI, S. *Android Architecture* [online], poslední revize 10.2.2012 [cit. 2016-11-22]. Dostupné z: http://java4freshers.blogspot.cz/2012/02/android-architecture.html

(28) MICROSOFT. *Get started with Surface RT or Surface 2* [online] [cit. 2016-11-17]. Dostupné z: https://www.microsoft.com/surface/en-us/support/getting-started/get-started-with-surface-rt

(29) MICROSOFT. *Getting started developing apps for Windows Phone 8 and Windows 8* [online] [cit. 2016-10-17]. Dostupné z: https://msdn.microsoft.com/cs-cz/enus/library/windows/apps/jj714071%28v=vs.105%29.aspx

(30) MICROSOFT. *Intro to the Universal Windows Platform* [online] [cit. 2016-11-17]. Dostupné z: https://msdn.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide

(31) MICROSOFT TECHNET. *Windows Mobile* [online] [cit. 2016-10-17]. Dostupné z: http://social.technet.microsoft.com/wiki/contents/articles/4380.windows-mobile.aspx

(32) MS POWERUSER. *Microsoft finally confirms Windows Phone 7 runs on Windows CE 7* [online] [cit. 2016-10-17]. Dostupné z: https://mspoweruser.com/microsoft-finally-confirms-windows-phone-7-runs-on-windows-ce-7/

(33) MS POWERUSER. *Still many more user interface and experience changes coming to Windows 10 Mobile* [online] [cit. 2016-11-17]. Dostupné z: https://mspoweruser.com/still-many-more-user-interface-and-experience-changes-coming-to-windows-10-mobile/

(34) MURPHY, M. *Android 2, Průvodce programováním mobilních aplikací.* Brno: Computer Press. 2011. 369s. ISBN 978-80-251-3194-7.

(35) NET APPLICATIONS. *Mobile and Tablet Operating System Market Share* [online] [cit. 2016-4-28]. Dostupné z: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1&qpf=1

(36) ORACLE CORPORATION. *Developer section for J2ME* [online] [cit. 2016-9-28]. Dostupné z: http://www.oracle.com/technetwork/java/javame/index.html

(37) ORACLE CORPORATION. *Programming the BlackBerry with J2ME* [online] [cit. 2016-10-5]. Dostupné z: http://www.oracle.com/technetwork/java/index-139239.html#2

(38) OSN *United nations projections* [online] [cit. 2016-9-28]. Dostupné z: https://esa.un.org/unpd/wpp/

(39) POCKET LINT. *Windows Phone 8: New hardware specs offer a new start* [online] [cit. 2016-10-17]. Dostupné z: http://www.pocket-lint.com/news/115972-windows-phone-8-hardware-specs

(40) SHAH, Vipul. *Open dialog to choose browser* [online] [cit. 2016-11-27]. Dostupné z: http://stackoverflow.com/questions/15017013/open-dialog-to-choose-browser

(41) SIEGLER, M. *Decoding Steve Jobs' Dressing Down Of Flash* [online] [cit. 2016-11-17]. Dostupné z: https://techcrunch.com/2010/04/29/steve-jobs-apple-adobe-flash/

(42) STATISTA INC. *Number of smartphone users worldwide* [online] [cit. 2016-9-28]. Dostupné z: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/

(43) TECH ADVISOR. *Windows Phone 8.1 and Cortana UK release date and new features* [online] [cit. 2016-10-17]. Dostupné z: http://www.pcadvisor.co.uk/news/mobile-phone/windows-phone-81-cortana-release-date-new-features-uk-3510108/

(44) THE INQUIRER. *Microsoft renames Metro to Modern UI* [online] [cit. 2016-10-17]. Dostupné z: http://www.theinquirer.net/inquirer/news/2198399/microsoft-renames-metro-to-modern-ui

(45) TOPLEY, K. *J2ME v kostce – Pohotová referenční příručka.* Praha: GRADA. 2004. 536s. ISBN 80-247-0426-9.

(46) UDGER. *UA list: Operating systems* [online] [cit. 2016-11-22]. Dostupné z: `https://udger.com/resources/ua-list/os`

(47) XAMARIN. *Xamarin: Mobile Application Development to Build Apps in C#* [online] [cit. 2016-11-20]. Dostupné z: `https://www.xamarin.com/`

(48) XAMARIN. *Xamarin Studio 6.0 release notes* [online] [cit. 2016-11-20]. Dostupné z: `https://developer.xamarin.com/releases/studio/xamarin.studio_6.0/xamarin.studio_6.0/`

Příloha **A**

Seznam použitých zkratek

ADT Android Development Tools

AOT Ahead-of-time

API Application Programming Interface

ARC Automatic Reference Counting

AVD Android Virtual Device

BES BlackBerry Enterprise Server

BIS BlackBerry Internet Service

BSSID Basic service set identifier

CLDC Connected Limited Device Configuration

C2DM Android Cloud to Device Messaging

GIT Global Information Tracker

HTML HyperText Markup Language

HTTP Hypertext Transfer Protocol

IMAP Internet Message Access Protocol

JDK Java Development Kit

JIT Just-in-time

J2EE Java 2 Enterprise Edition

J2ME Java 2 Micro Edition

J2SE Java 2 Standard Edition

MAC Media access control address

MIDP Mobile Information Device Profile

MMS Multimedia Messaging Service

MP3 MPEG Audio Layer III

NDK Native Development Kit

NFC Near Field Communication

OS operační systém

OSN Organizace spojených národů

PC Personal computer

PDA Personal Digital Assistant

POP Post Office Protocol

RSSI Received signal strength indicator

SDK Software development kit

SMTP Simple Mail Transfer Protocol

SQL Structured Query Language

SSL Secure Sockets Layer

SVN Apache Subversion

URL Uniform Resource Locator

USB Universal Serial Bus

USD United States dollar

voIP Voice over IP

WVGA Wide Video Graphics Array

WWDC Worldwide Developer Conference

XHTML Extensible Hypertext Markup Language

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `Android` ..... adresář obsahující Android aplikaci Sensors Toolbox
  - `bin` ..... adresář se spustitelnou verzí pro OS Android
  - `sources` ..... Eclipse projekt se zdrojovými kódy pro OS Android
- `text` ..... text práce
  - `DP_Becka_Michal_2017.pdf` ..... text práce ve formátu PDF